TaIwan Multi-scale Community Ocean Model (TIMCOM) User's Manual

Jay Chih-Chieh Young, Yu-Heng Tseng,
Mao-Lin Shen, Yu-Chiao Liang,
Mu-Hua Chien, & Chia-Hung Chien
Copyright © *Draft date February 20, 2012*

Preface

The objective of this document is to introduce TaIwan Multi-scale Community Ocean Model (TIMCOM) which is an accurate, efficient and user-friendly framework to simulate real or idealized ocean flows over wide-ranging scales. This manual is organized as follows: Chapter 1 briefly describes the development history and model features (as well as ongoing improvement); Chapter 2 gives the governing equations and numerical methods; Chapter 3 expresses the grid coupling approach; Chapter 4 presents the code structure; Chapter 5 tells how to run the model. Readers who have background knowledge of fluid mechanics, computational fluid dynamics, and Fortran programming language shall have no problem following this manual.

Abstract

The TaIwan Multi-scale Community Ocean Model (TIMCOM) is developed to provide an accurate, efficient, and user-friendly framework for studying a variety of ocean flows over a wide-ranging scales and boundary conditions. The model employs the finite volume concept and discretizes the governing equations with fourth-order approximation in space and modified Leap-frog temporal scheme. The resulting system matrix can be directly solved by an efficient error vector propagation (EVP) elliptic solver. Grid coupling technique is adopted for the problems with complicated geometry, giving high accuracy in the required domain and greatly reduced computational cost. Several benchmark tests and real applications, which can be easily customized through a user interface, are used to examine the models capability. Robustness, efficiency, and accuracy of TIMCOM are demonstrated and analyzed.

Keywords: TIMCOM, community model, fourth order, EVP solver, grid coupling

Contents

1 Introduction **1**

2 Model Description **4**
2.1 Governing Equations and Boundary Conditions 4
2.1.1 Governing Equations 4
2.1.2 Boundary Conditions 5
2.2 Numerical Methods 6

3 Grid Coupling Algorithm **10**
3.1 TAI to NPB 11
3.2 NPB to TAI 11

4 Code Organization **13**
4.1 Program Directories 13
4.2 Source Codes 16
4.2.1 Variable Modules 16
4.2.2 Preprocessor 16
4.2.3 Main Code 17
4.2.4 Postprocessor 17

5 Model Applications and Tutorials **19**
5.1 Obtain the Code 19
5.2 2D Island Wakes 20
5.2.1 Description 20
5.2.2 Running the Model and Plotting the Results 20
5.3 3D Density Current over a Continental Slope 23
5.3.1 Description 23
5.3.2 Running the Model and Plotting the Results 24
5.4 Global Ocean Modelling 26
5.4.1 Description 26
5.4.2 Running the Model and Plotting the Results 26
5.5 Dual-domain North Pacific Ocean Modelling 29
5.5.1 Description 29
5.5.2 Running the Model and Plotting the Results 29

A Nonuniform Vertical Grid Spacing **33**

B Vertical Mixing Scheme: Pacanowski & Philander (1981) 35

C Fourth-Order Accurate Interpolation from Nearby Control Volume Averages to a Cell Face 36

D Error Vector Propagation Method 39

E Listing of Variables 41

Chapter 1

Introduction

TaIwan Multi-scale Community Ocean Model (TIMCOM) is intended to provide a user-friendly and powerful framework for simulating real or idealized ocean flows over wide-ranging scales and boundary conditions. Figure 1.1 shows the family tree of TIMCOM and the history of ocean general circulation models. The DieCAST[1] (Dietrich Center for Air Sea Technology) ocean model, an earlier work in TIMCOM family, has been well validated over the past decade(Dietrich & Ko, 1994; Dietrich, 1997; Dietrich & Lin, 2002; Tseng *et al.*, 2005). While robust and accurate for research/applications, the DieCAST model was not easily (or widely) used in the ocean community due to difficult readability, complicated coding structure, tedious customization process, etc. As a consequence, the development of TIMCOM is motivated.

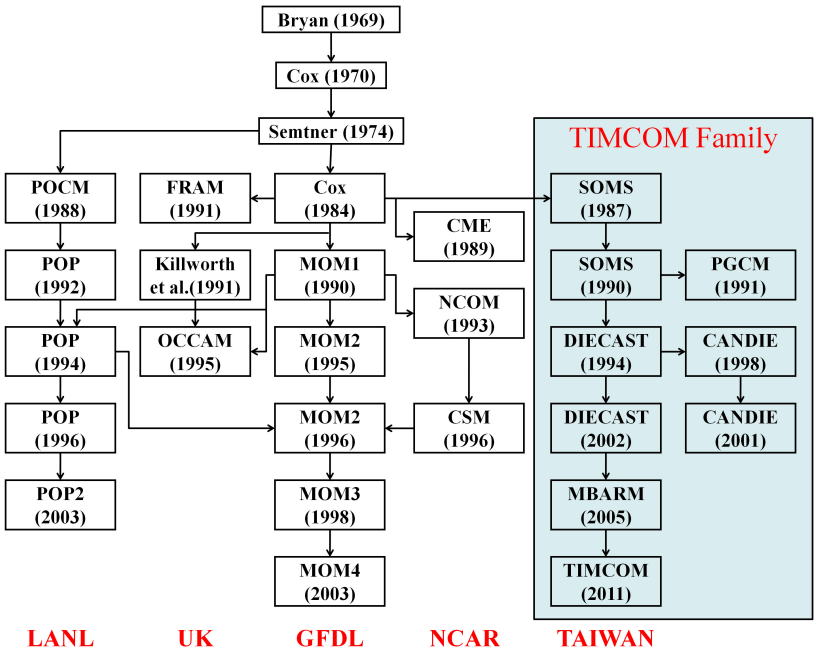

Figure 1.1: The family tree of TIMCOM and development history of ocean models.

[1]Reader who are interested in detailed description and user's guide of the DieCAST model can refer to the website *http://efdl.as.ntu.edu.tw/research/diecast/index.html* (Canadian version, i.e. CANDIE, is also available via *http://www.phys.ocean.dal.ca/programs/CANDIE/*)

TIMCOM is developed basically based upon the DieCAST model with some modifications (e.g. high-order temporal filter) and extensions (e.g. the implementation of particle tracking). Written in Fortran 90 programming language with a modular approach, TIMCOM has clear control flow and better coding layout. A friendly user interface is further introduced, making user customization easy. In addition, model output adopts the Network Common Data Form (NetCDF), which allows machine independent access and sharing of data. The newly developed TIMCOM is a robust, efficient, accurate, and user-friendly model for studying ocean flow problems. Major features of TIMCOM are summarized as follows.

- Hydrostatic or non-hydrostatic pressures:
  TIMCOM solves the unsteady, incompressible, 3D primitive equations with Boussinesq and hydrostatic approximations on a full spherical coordinate. For the applications with rapidly changed bathymetry or density inversion, non-hydrostatic version (Dietrich & Lin, 2002) is required and available under licensing agreement.

- Rigid lid or free surface:
  TIMCOM adopts a rigid-lid approximation to efficiently and effectively resolve general circulation problems. For high-frequency free-surface applications, explicit time-split or semi-implicit free surface versions are recommended (available under license).

- Nonlinear equation of state:
  A standard nonlinear equation of state is used to calculate in-situ density in terms of potential temperature, salinity and pressure-depth.

- Turbulence mixing (Viscosity & Diffusivity):
  Weakly physical-based parameterization is used for the horizontal mixing. For the vertical mixing, TIMCOM employs the approach of Pacanowski & Philander (1981), instead of computation- and storage-intensive Mellor-Yamada level 2.5 turbulence closure in SOMS model (Dietrich *et al.*, 1987). The users can also apply any preferred scheme.

- Non-damping surface nudging:
  An precise surface flux treatment (Dietrich *et al.*, 2004b) which does not damp transients and gives rapid convergence to ensemble average annual cycle climatology is utilized. It may be used in combination with synoptic wind and surface flux forcing.

- Highly-accurate and low-dissipative numerical scheme:
  TMCOM uses a blend of collocated and staggered grid structure with non-uniform grid increments. Second-order Leap-frog temporal scheme and fourth-order-accurate spatial discretization (or interpolation) are applied to discretize the governing equations. Further, modified Robert-Asselin filter with high accuracy are employed.

- Efficient pressure Poisson solver:
  Efficient and accurate Error Vector Propagation (EVP) solver is used to solve the system matrix. EVP solver is also ideal and suitable for parallel computing (Tseng & Chien, 2011).

- Flux-conserved grid coupling algorithm:
  Simple, efficient, stable, and flux-conserved grid coupling algorithm (Dietrich *et al.*, 2004b) facilitates multi-scale dynamics modeling with a greatly reduced computational cost.

- Particle tracking:
  TIMCOM now has the capability to release tracers from a specified location for particle tracking.

The hydrostatic, rigid-lid approximation version of TIMCOM is now released and available on the website (*http://efdl.as.ntu.edu.tw/research/timcom/index.html*). The following chapters will briefly express the governing equations, numerical methods, grid coupling algorithm, code structure, and the model execution. Currently, several advanced features such as (i) immersed boundary methods for topography, (ii) turbulence parameterization options, (iii) hindcast wind forcing capability, (iv) ensemble simulation capability, (v) bio-geochemistry dynamics are under development and will be reported in the near future.

Chapter 2

Model Description

2.1 Governing Equations and Boundary Conditions

2.1.1 Governing Equations

The governing equations are the unsteady, incompressible, three-dimensional (3D) primitive equations with the Boussinesq, and hydrostatic approximation. These equations in the spherical coordinate (λ, ϕ, z) and time t can be written as

- Continuity equation:

$$\frac{1}{R\cos\phi}\left(\frac{\partial u}{\partial \lambda}+\frac{\partial (v\cos\phi)}{\partial \phi}\right)+\frac{\partial w}{\partial z}=0, \tag{2.1}$$

- Horizontal momentum equations:

$$\frac{\partial u}{\partial t}+\mathcal{L}u-\left(f+\frac{u\tan\phi}{R}\right)v=-\frac{1}{\rho_0 R\cos\phi}\frac{\partial p}{\partial \lambda}+\mathcal{D}_m u+\frac{\partial}{\partial z}\left(A_u\frac{\partial u}{\partial z}\right), \tag{2.2}$$

$$\frac{\partial v}{\partial t}+\mathcal{L}v+\left(f+\frac{u\tan\phi}{R}\right)u=-\frac{1}{\rho_0 R}\frac{\partial p}{\partial \phi}+\mathcal{D}_m v+\frac{\partial}{\partial z}\left(A_v\frac{\partial v}{\partial z}\right), \tag{2.3}$$

- Hydrostatic equation:

$$\frac{\partial p}{\partial z}=-\rho g, \tag{2.4}$$

- Equation of state:

$$\rho=\rho(S,T,p), \tag{2.5}$$

- Conservation of scalar (potential temperature and salinity)

$$\frac{\partial T}{\partial t}+\mathcal{L}T=\mathcal{D}_h T+\frac{\partial}{\partial z}\left(K_T\frac{\partial T}{\partial z}\right), \tag{2.6}$$

$$\frac{\partial S}{\partial t}+\mathcal{L}S=\mathcal{D}_h S+\frac{\partial}{\partial z}\left(K_S\frac{\partial S}{\partial z}\right), \tag{2.7}$$

where u, v, and w are the velocity components in the longitudinal λ, latitudinal ϕ, and vertical z directions, respectively; p is pressure and $p = p_s + p_b$, where p_s represents the surface pressure and baroclinic pressure p_b is

$$p_b = g \int_z^0 \rho dz; \tag{2.8}$$

ρ is the in-situ density and ρ_0 is the reference density; T and S are potential temperature and salinity, respectively; A_u and A_v are the vertical eddy viscosity coefficients for horizontal momentum equations while K_T and K_S are the vertical eddy diffusivity for temperature and salinity equations, respectively; f is the Coriolis parameter; g is the gravitational acceleration; the convection operator is

$$\mathcal{L} = \frac{u}{R\cos\phi}\frac{\partial}{\partial\lambda} + \frac{v}{R}\frac{\partial}{\partial\phi} + w\frac{\partial}{\partial z}; \tag{2.9}$$

and the horizontal diffusion operator is

$$D_{m(h)} = \frac{A_{m(h)}}{R^2}\left(\frac{1}{\cos^2\phi}\frac{\partial^2}{\partial\lambda^2} - \tan\phi\frac{\partial}{\partial\phi} + \frac{\partial^2}{\partial\phi^2}\right), \tag{2.10}$$

where A_m and A_h are the horizontal eddy viscosity and diffusivity, respectively.

2.1.2 Boundary Conditions

Various boundary conditions are used for solving the governing equations. At the sea surface $z = 0$, the rigid-lid approximation can be used when surface waves are not considered, giving $w_{z=0} = 0$. At the bottom $z = -h$, the boundary condition is

$$\frac{1}{R\cos\phi}\left(u\frac{\partial h}{\partial\lambda} + v\cos\phi\frac{\partial h}{\partial\phi}\right) + w_{z=-h} = 0. \tag{2.11}$$

By integrating the continuity equation (2.1) over the water depth and applying Leibniz's rule with the boundary conditions mentioned above, we obtain the conservative form of continuity equation, i.e.

$$\frac{1}{R\cos\phi}\left[\frac{\partial}{\partial\lambda}\left(\int_{-h}^0 u dz\right) + \frac{\partial}{\partial\phi}\left(\int_{-h}^0 v\cos\phi dz\right)\right] = 0. \tag{2.12}$$

At the inflow boundary, the normal velocity component can be specified based upon available measurements. At the outflow boundary, the Orlanski radiation condition is employed, i.e.

$$\frac{\partial\phi}{\partial t} = -U_n\frac{\partial\phi}{\partial n}, \tag{2.13}$$

where ϕ represents velocity components, temperature or salinity; U_n is the normal velocity on the open boundary;

$$\frac{\partial\phi}{\partial n} = \begin{cases} (\phi - \phi_o)/\Delta x_n & U_n \geq 0, \\ (\phi_i - \phi)/\Delta x_n & U_n < 0, \end{cases} \tag{2.14}$$

the location suffix o indicates the open boundary while i means one-grid-point inside; and Δx_n is the grid spacing in the direction normal to the boundary. For solid walls, the normal velocity is set to zero, i.e. the impermeable condition, and the tangential velocities are determined by either non-slip or free-slip condition.

2.2 Numerical Methods

TIMCOM is an efficient and accurate model using the second-order Leap-frog temporal scheme with fourth-order spatial approximation. The modified Robert-Asselin-Williams filter is applied to eliminate the time splitting issue. The model domain is discretized by a set of resolution parameters, i.e. I_0, J_0, and K_0 along the horizontal and vertical directions with the grid index i, j, and k, respectively (see Fig. 2.1). The setting of horizontal grid spacing is shown in Fig. 2.2. Non-uniform vertical layers can be employed (see Appendix A). For variable arrangement, TIMCOM uses a blend of collocated and staggered grids structures (Arakawa A and C grids). The cell-averaged variables $u_{i,j,k}$, $v_{i,j,k}$, $S_{i,j,k}$, $T_{i,j,k}$, $p_{i,j,k}$, and $\rho_{i,j,k}$ are defined at the cell center (i, j, k), while the face-averaged velocity components $U_{i+1/2,j,k}$, $V_{i,j+1/2,k}$, and $W_{i,j,k+1/2}$ are located on the faces $(i + 1/2, j, k)$, $(i, j + 1/2, k)$, and $(i, j, k + 1/2)$. Note that $\rho_{i,j,k}$ in the model represents density variations ρ', which is a small deviation from the mean reference value ρ_0, i.e. $\rho' = \rho - \rho_0$. As a consequence, $p_{i,j,k}$ indicates pressure perturbations, rather than the total pressures.

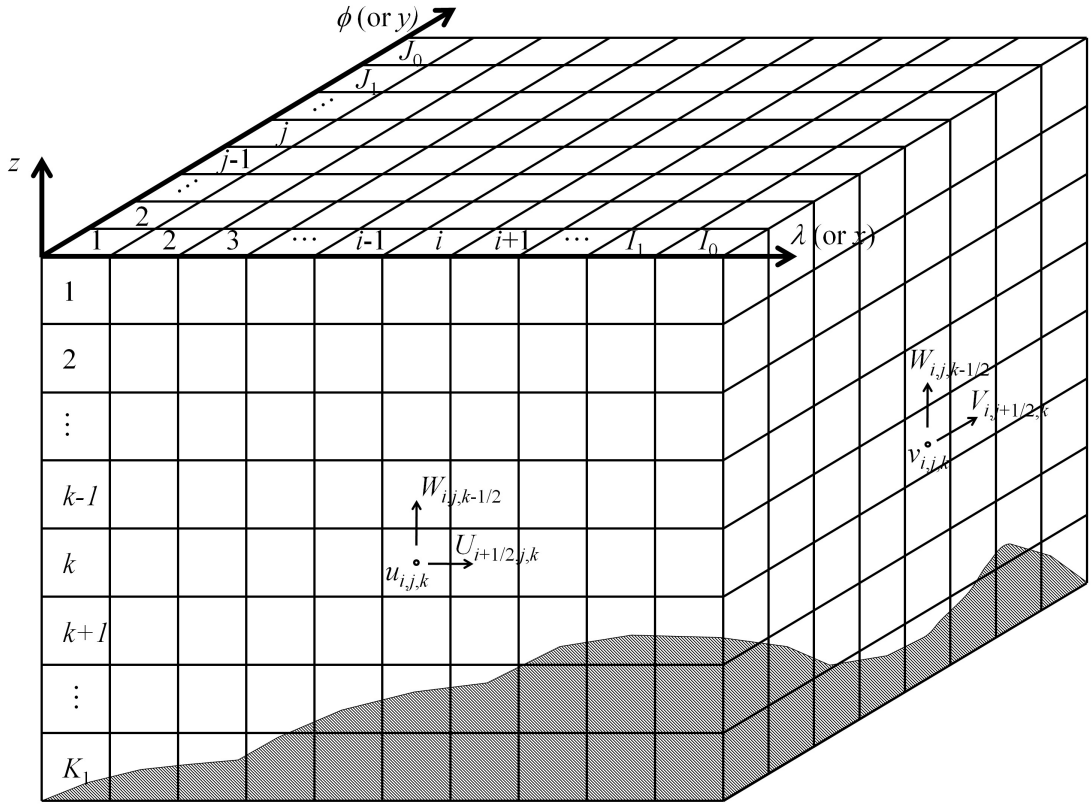

Figure 2.1: computational grids in TIMCOM

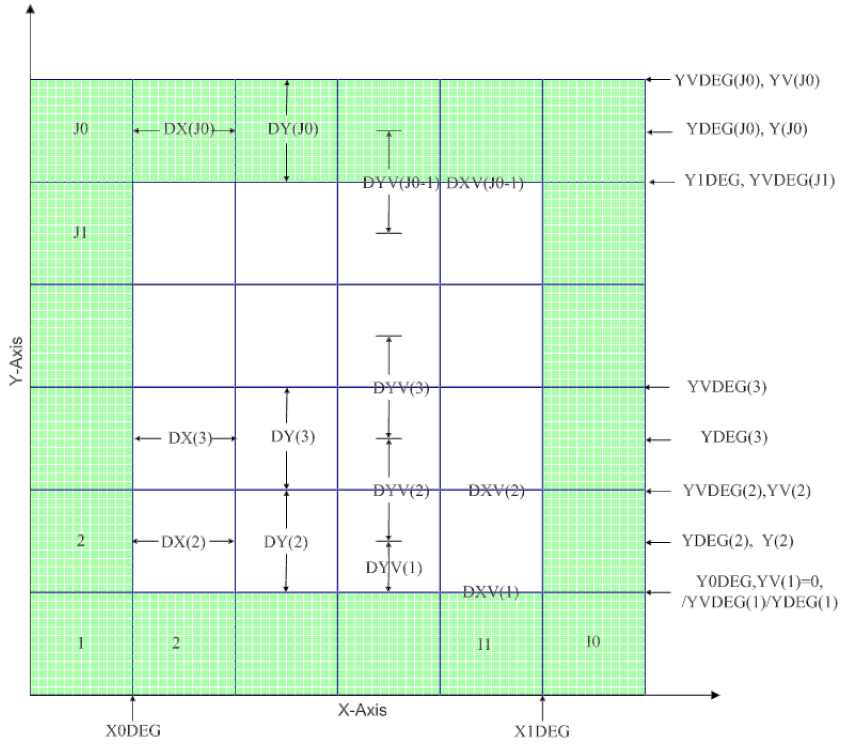

Figure 2.2: horizontal grid spacings of the computational grid

The overall numerical algorithm of TIMCOM can be divided into three major steps: (i) update intermediate flow field through horizontal momentum equations (2.2)-(2.3) as well as conservation equations (2.6)-(2.7) at the time step n; (ii) substitute the intermediate velocities and corrections into the conservative continuity equation (2.12) and solve the resulting 2D Poisson equation for pressure adjustment. Then, the final flow field can be readily determined; (iii) apply modified Robert-Asselin filter to the flow variables and advance to the time step $n+1$. Each step is further explained in the following.

In the first step, we discretize the horizontal momentum equation (2.2) to calculate the intermediate longitudinal-direction velocity $\tilde{u}^{n+1}$, i.e.

$$\frac{\tilde{u}^{n+1}_{i,j,k} - u^{n-1}_{i,j,k}}{2\Delta t} = -\mathcal{L}u^n - \frac{1}{\rho_0 R\cos\phi}\frac{\partial\left(p_s^{n-1}+p_b^n\right)}{\partial\lambda} + \mathcal{D}_m u^{n-1} + \frac{\partial}{\partial z}\left[A_u\frac{\partial}{\partial z}\left(u^{n-1}\right)\right], \quad (2.15)$$

where

$$\begin{aligned}\mathcal{L}u^n &= \frac{U^n_{i+1/2,j,k}u^n_{i+1/2,j,k} - U^n_{i-1/2,j,k}u^n_{i-1/2,j,k}}{R\cos\phi\Delta\lambda_i} \\ &+ \frac{V^n_{i,j+1/2,k}u^n_{i,j+1/2,k} - V^n_{i,j-1/2,k}u^n_{i,j-1/2,k}}{R\Delta\phi_j} \\ &+ \frac{W^n_{i,j,k+1/2}u^n_{i,j,k+1/2} - W^n_{i,j,k-1/2}u^n_{i,j,k-1/2}}{\Delta z_k}, \end{aligned} \quad (2.16)$$

$$\frac{\partial}{\partial\lambda}\left[p^{n-1(n)}_{s(b)}\right] = \frac{\left[p_{s(b)}\right]^{n-1(n)}_{i-2,j,k} - 8\left[p_{s(b)}\right]^{n-1(n)}_{i-1,j,k} + 8\left[p_{s(b)}\right]^{n-1(n)}_{i+1,j,k} - \left[p_{s(b)}\right]^{n-1(n)}_{i+2,j,k}}{12\Delta\lambda_i}, \quad (2.17)$$

$$\mathcal{D}_m u^{n-1} = \frac{A_m}{R^2}\left(\frac{u^{n-1}_{i+1,j,k} - 2u^{n-1}_{i,j,k} + u^{n-1}_{i-1,j,k}}{\cos^2\phi\Delta\lambda_i} + \frac{u^{n-1}_{i,j+1,k} - 2u^{n-1}_{i,j,k} + u^{n-1}_{i,j-1,k}}{\Delta\phi_j} + \tan\phi\frac{u^{n-1}_{i,j+1,k} - u^{n-1}_{i,j-1,k}}{2\Delta\phi_i}\right), \tag{2.18}$$

$$\frac{\partial}{\partial z}\left[A_u\frac{\partial}{\partial z}\left(u^{n-1}\right)\right] = \frac{1}{\Delta z_k}\left[(A_u)_{i,j,k}\frac{u^{n-1}_{i,j,k+1} - u^{n-1}_{i,j,k}}{\Delta z_k} - (A_u)_{i,j,k-1}\frac{u^{n-1}_{i,j,k} - u^{n-1}_{i,j,k-1}}{\Delta z_k}\right], \tag{2.19}$$

in which the vertical eddy viscosity A_u is determined by the approach of Pacanowski & Philander (1981) (see Appendix B). Similar procedure can be directly applied to obtain the intermediate latitudinal-direction velocity $\tilde{v}^{n+1}$, final potential temperature T^{n+1} and salinity S^{n+1}. The intermediate velocities are further updated to involve the Coriolis effects, i.e.

$$\hat{u}^{n+1} = \tilde{u}^{n+1} + 2\Delta t\tilde{F}(\frac{\hat{v}^{n+1} + v^{n-1}}{2}) = \tilde{u}^{n+1} + \tilde{F}\Delta t v^{n-1} + \tilde{F}\Delta t\hat{v}^{n+1}, \tag{2.20}$$

$$\hat{v}^{n+1} = \tilde{v}^{n+1} - 2\Delta t\tilde{F}(\frac{\hat{u}^{n+1} + u^{n-1}}{2}) = \tilde{v}^{n+1} - \tilde{F}\Delta t u^{n-1} - \tilde{F}\Delta t\hat{u}^{n+1}, \tag{2.21}$$

where $\tilde{F} \equiv f + \frac{u^n\tan\phi}{R}$. Solving equations (2.20) and (2.21) gives

$$\hat{u}^{n+1} = \frac{1}{1 + (\tilde{F}\Delta t)^2}\left(\hat{u}^{n+1}_* + \tilde{F}\Delta t\cdot\hat{v}^{n+1}_*\right), \tag{2.22}$$

$$\hat{v}^{n+1} = \frac{1}{1 + (\tilde{F}\Delta t)^2}\left(\hat{v}^{n+1}_* - \tilde{F}\Delta t\cdot\hat{u}^{n+1}_*\right), \tag{2.23}$$

where

$$\hat{u}^{n+1}_* = \tilde{u}^{n+1} + \tilde{F}\Delta t\cdot v^{n-1}, \tag{2.24}$$

$$\hat{v}^{n+1}_* = \tilde{v}^{n+1} - \tilde{F}\Delta t\cdot u^{n-1}. \tag{2.25}$$

For the face-averaged velocity field, the fourth-order interpolation (Sanderson & Brassington, 1998) is employed (see Appendix C for details). The intermediate face velocity is given by

$$\hat{U}^{n+1}_{i+1/2,j,k} = \frac{-\hat{u}^{n+1}_{i-1,j,k} + 7\hat{u}^{n+1}_{i,j,k} + 7\hat{u}^{n+1}_{i+1,j,k} - \hat{u}^{n+1}_{i+2,j,k}}{12}. \tag{2.26}$$

The second step is to determine the final velocity field. The basic idea is that correct the intermediate velocities to achieve an appropriate pressure formulation, i.e. p^n_s rather than p^{n-1}_s used in equation (2.15). Thus, the final face velocities can be written as

$$U^{n+1}_{i+1/2,j,k} = \hat{U}^{n+1}_{i+1/2,j,k} + \delta U^{n+1}_{i+1/2,j,k}, \tag{2.27}$$

$$V^{n+1}_{i,j+1/2,k} = \hat{V}^{n+1}_{i,j+1/2,k} + \delta V^{n+1}_{i,j+1/2,k}, \tag{2.28}$$

where

$$\delta U^{n+1}_{i+1/2,j,k} = -\frac{2\Delta t}{\rho_0 R\cos\phi}\frac{\partial}{\partial\lambda}\left(p^n_s - p^{n-1}_s\right) = -\frac{2\Delta t}{\rho_0 R\cos\phi}\frac{\partial}{\partial\lambda}\left(\delta p^n_s\right), \tag{2.29}$$

$$\delta V^{n+1}_{i,j+1/2,k} = -\frac{2\Delta t}{\rho_0 R}\frac{\partial}{\partial\phi}\left(p^n_s - p^{n-1}_s\right) = -\frac{2\Delta t}{\rho_0 R}\frac{\partial}{\partial\phi}\left(\delta p^n_s\right). \tag{2.30}$$

Substituting equations (2.27)-(2.30) into the conservative continuity equation (2.12) results in the 2D Poisson equation for pressure adjustment δp_s^n, i.e.

$$\frac{2\Delta t}{\rho_0 R^2 \cos\phi}\int_{-h}^{0}\left[\frac{\partial}{\partial\lambda}\frac{1}{\cos\phi}\frac{\partial\delta p_s^n}{\partial\lambda}+\frac{\partial}{\partial\phi}\left(\frac{\partial\delta p_s^n}{\partial\phi}\right)\cos\phi\right]dz=\hat{w}_{-h}^{n+1}, \tag{2.31}$$

where the last term is a convenient shorthand notation and can be expressed as

$$\hat{w}_{-h}^{n+1}=\frac{1}{R\cos\phi}\left(\frac{\partial}{\partial\lambda}\int_{-h}^{0}\hat{U}^{n+1}dz+\frac{\partial}{\partial\phi}\int_{-h}^{0}\hat{V}^{n+1}\cos\phi dz\right). \tag{2.32}$$

The system equation (2.31) can be solved by an efficient error vector propagation (EVP) elliptic solver (see Appendix D). Once the pressure adjustment δp_s^n is determined, the corrections for the face-averaged velocities $\delta U_{i+1/2,j,k}^{n+1}$ and $\delta V_{i,j+1/2,k}^{n+1}$ can be directly obtained by equations (2.29)-(2.30). Further, interpolate the velocity corrections back to the collocated point (i,j,k), giving the changes for the final cell-averaged velocities, e.g.

$$u_{i,j,k}^{n+1}=\hat{u}_{i,j,k}^{n+1}+\delta u_{i,j,k}^{n+1}, \tag{2.33}$$

where

$$\delta u_{i,j,k}^{n+1}=\frac{-\delta U_{i-3/2,j,k}^{n+1}+13\delta U_{i-1/2,j,k}^{n+1}+13\delta U_{i+1/2,j,k}^{n+1}-\delta U_{i+3/2,j,k}^{n+1}}{24}. \tag{2.34}$$

The flow variables u^{n+1}, v^{n+1}, U^{n+1}, V^{n+1}, and p_s^n are fully updated by velocity corrections and pressure adjustment. Vertical velocity W^{n+1} is calculated using continuity equation (2.1).

In the last step, the modified Robert-Asselin filter is applied. Recognizing that the Leap-frog temporal scheme is widely used in many general circulation models, Williams (2009) proposed a modified filter to resolve the time slitting issue. In contrast to the standard Robert-Asselin filter, the modified version provides third-order accuracy for amplitude errors. The filtered velocities for the next time step $t=t+\Delta t$ are

$$u_{i,j,k}^{n-1}\Big|_{t=t+\Delta t} = \left[u_{i,j,k}^{n}+\frac{\upsilon\alpha}{2}\left(u_{i,j,k}^{n-1}-2u_{i,j,k}^{n}+u_{i,j,k}^{n+1}\right)\right]\Big|_{t=t}, \tag{2.35}$$

$$u_{i,j,k}^{n}\Big|_{t=t+\Delta t} = \left[u_{i,j,k}^{n+1}-\frac{\upsilon(1-\alpha)}{2}\left(u_{i,j,k}^{n-1}-2u_{i,j,k}^{n}+u_{i,j,k}^{n+1}\right)\right]\Big|_{t=t}, \tag{2.36}$$

where $0\le\upsilon\le 1$ and $0\le\alpha\le 1$ are two filter parameters controlling the family of the modified Robert-Asselin-filtered Leap-frog scheme (denoted as MRAF). Note that the special case $\upsilon=0$ leads to the classic (non-filtered) Leap-frog scheme (denoted as NONF) while the case $\alpha=1$ results in the standard Robert-Asselin-filtered Leap-frog scheme (denoted as SRAF).

Chapter 3

Grid Coupling Algorithm

Efficient and accurate ocean modeling is desirable in practical problems. Due to the complicated geometries (topography) or the restriction in computation resources, it is usually difficult to efficiently obtain satisfactory solutions only using a single grid. TIMCOM has the capability of resolving a problem on dual or more grids with different resolutions. Based upon the grid coupling technique which combines grids through data transferring, the required accuracy in each domain can be achieved and the overall computational cost are greatly reduced.

The algorithm for grid coupling must ensure flux conservation, stability, efficiency, and simplicity. To reach this purpose, linear interpolation and time filtering approach are utilized in TIMCOM. Here, we use the North Pacific Ocean and Taiwan area modelling (Tseng *et al.*, 2011) to demonstrate the grid coupling algorithm. The computational domain ranges from 100°E to 80°W and from 30°S to 60°N (see Figure 3.1). While a finer horizontal resolution of 1/8° is used in the TAI domain (i.e. west of 150°E, left red block in Figure 3.1) to accurately resolve the detailed Kuroshio and the associated circulations in the marginal seas, a 1/4° horizontal resolution is used in the NPB domain (i.e. east of 150°E, right blue block in Figure 3.1) to reduce the computational cost. Two grids are coupled by overlapping boundary cell(s), i.e. the grid index $i = 2$ in the NPB domain covers the grid indices $i = I_2$ and $i = I_1$ in the TAI domain. Note that the southern boundary cells, i.e. $j = 2$ and $j = 3$, in the TAI domain correspond to the particular latitudinal cell $j = 127$ in the NPB domain. In the following, we briefly present the grid coupling algorithm. One can refer to Dietrich *et al.* (2004b) for details.

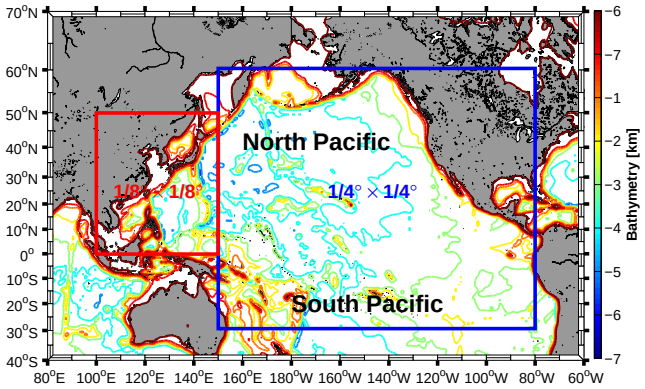

Figure 3.1: Computational domain of the North Pacific Ocean and Taiwan area modelling.

3.1 TAI to NPB

We demonstrate how to transfer values from the TAI domain to the NPB domain (see Figure 3.2 (a)). Both face and center variables are considered. In the following, only horizontal velocity component U (or u) is illustrated for the purpose of brevity.

Cell Face Variables

For the fluid cells, the horizontal velocity at the west interface of NPB domain (blue arrow), e.g. $(U_{1+1/2,127,k})_{NPB}$, can be obtained by averaging the horizontal velocities in the TAI domain (red arrows), e.g. $(U_{I_3+1/2,2,k})_{TAI}$ and $(U_{I_3+1/2,3,k})_{TAI}$. However, due to the difference of the time increments ($\Delta t_{TAI} < \Delta t_{NPB}$), the interpolated velocity may cause stability problems. To avoid this issue, the commonly used time filtering technique is applied, i.e.

$$(U^{n+1}_{1+1/2,127,k})_{NPB} = f_1 \cdot \frac{1}{2} \cdot \left[(U^{n+1}_{I_3+1/2,2,k})_{TAI} + (U^{n+1}_{I_3+1/2,3,k})_{TAI}\right] + f_2 \cdot (U^{n}_{1+1/2,127,k})_{NPB}, \quad (3.1)$$

where f_1 and $f_2 = 1 - f_1$ are the time filtering parameters.

Cell Center Variables

To improve the stability, the current cell-centered horizontal velocity in the NPB domain is updated by a smooth increment δu, which is the difference between the TAI interpolated velocity and the previous-time-step value. The cell-centered horizontal velocity, e.g. $(u_{2,369,k})_{NPB}$, can be expressed as

$$(u^{n+1}_{2,369,k})_{NPB} = (u^{n}_{2,369,k})_{NPB} + \delta u, \quad (3.2)$$

where

$$\delta u = \frac{1}{4} \cdot \left[(u^{n+1}_{I_2,J_2,k})_{TAI} + (u^{n+1}_{I_2,J_1,k})_{TAI} + (u^{n+1}_{I_1,J_2,k})_{TAI} + (u^{n+1}_{I_1,J_1,k})_{TAI}\right] - (u^{n-1}_{2,369,k})_{NPB}. \quad (3.3)$$

3.2 NPB to TAI

Subsequently, we present the procedure to transfer values from the NPB domain to the TAI domain (see Figure 3.2 (b)). The grid size in the NPB domain is twice as large as that in the TAI domain so one NPB cell can completely cover four TAI cells. Based upon this fact, we use a quick and simple way to transfer date, i.e. the NPB value and its surrounding TAI values are the same for both face and center variables.

Cell Face Variables

For the fluid cells, the horizontal velocities at the east interface of TAI domain simply take the value of NPB horizontal velocities. Particularly, no temporal filtering is required due to the smaller time step in the TAI domain. The TAI face velocities, e.g. $(U_{I_1+1/2,2,k})_{TAI}$ and $(U_{I_1+1/2,3,k})_{TAI}$, are written as

$$(U^{n+1}_{I_1+1/2,2,k})_{TAI} = (U^{n+1}_{I_1+1/2,3,k})_{TAI} = (U^{n+1}_{2+1/2,127,k})_{NPB}. \quad (3.4)$$

Cell Center Variables

For the TAI domain, the horizontal velocities at the east boundary ghost zone cell ($I = I_0$) are exactly equal to the NPB horizontal velocity, e.g.

$$(u^{n+1}_{I_0,J_2,k})_{TAI} = (u^{n+1}_{I_0,J_1,k})_{TAI} = (u^{n+1}_{3,369,k})_{NPB}. \quad (3.5)$$

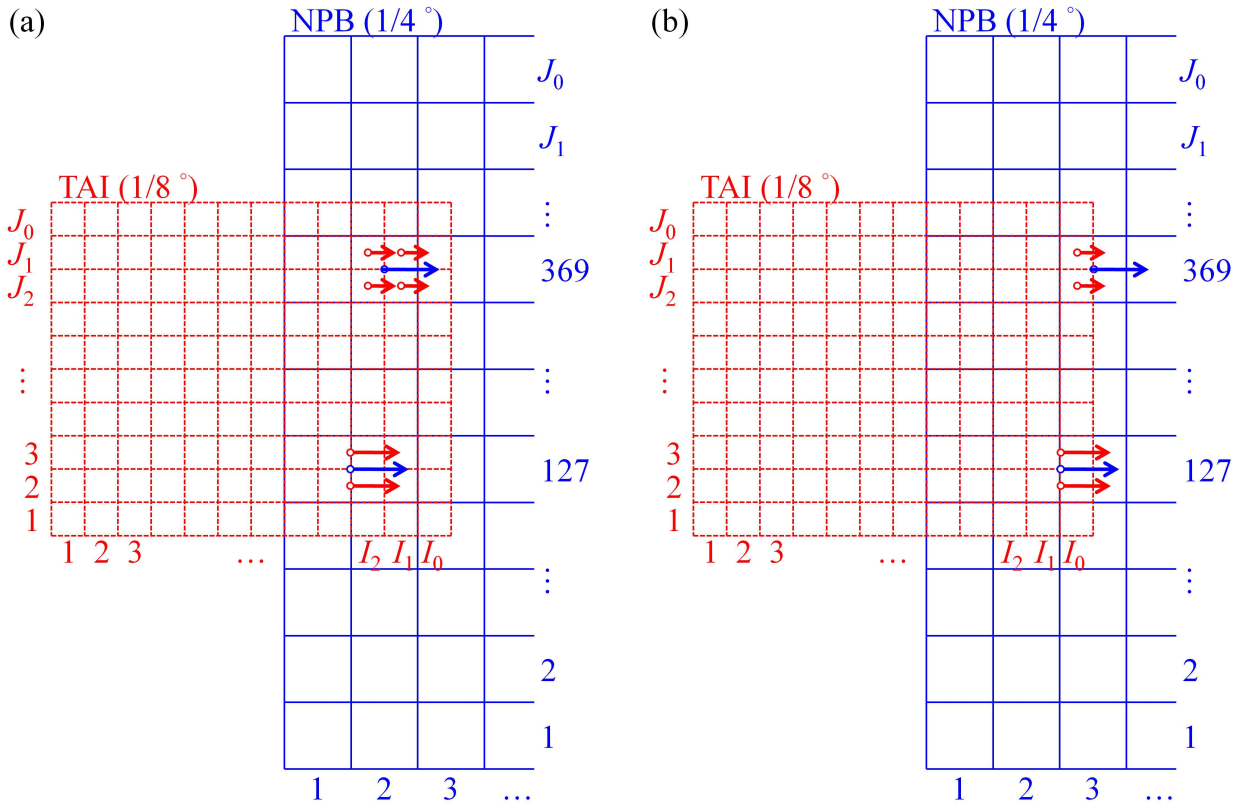

Figure 3.2: Data transferring (a) from TAI to NPB and (b) from NPB to TAI.

Chapter 4

Code Organization

TIMCOM is written in Fortran 90 programming language with a modular approach, which makes future modification (or improvement) of the code simple and easy. TIMCOM supports Intel and pgi compilers and can work on Windows or Unix system. TIMCOM contains three major components. They are (i) **Preprocessor**, (ii) **Main Code**, and (iii) **Postprocessor**. Their associated program directories and source codes will be presented in this chapter.

4.1 Program Directories

Figure 4.1 shows the detailed program directories in TIMCOM. The major directories under the root TIMCOM are DOC, DATA, PREP, MAIN, and POST. Brief description for each directory is given in the following.

DOC :
The introductory document, i.e. this user's manual (TIMCOM_Users_Manual.pdf), exists in this directory.

DATA :
DATA is a link directed to the default (or user-specified) location where the necessary data for TIMCOM simulation is saved (The default location is "username/DATA"). Note that three types of data are generally required. They are (i) bathymetry, (ii) temperature & salinity, and (iii) wind stress. Subdirectories are further explained.

- Etopo:
  Data files in Etopo directory provide the bathymetry/bottom topography elevations on a 5-minute grid resolution. TIMCOM uses the "5-Minute Gridded Global Relief Data Collection" which is maintained by the National Geophysical Data Center of National Oceanic & Atmospheric Administration.
- Levitus:
  Levitus indicates the Levitus climatology data of the Ocean Climate Laboratory of the National Oceanographic Data Center. Annual, seasonal, and monthly long-term means for temperature and salinity at different depths are contained in the data files.

- Hellerman:
  Hellerman means Hellerman and Rosenstein Global Wind Stress Climatology. The data files include monthly averaged wind stress over the global ocean from the year 1870 to 1976.

PREP :

PREP stands for the preprocessor system. In the PREP directory, a template directory which contains all necessary subdirectories and files can be seen after the installation of TIMCOM. Once a case is specified (using the prep_setup command), a directory with the corresponding CASENAME, e.g. ILE2D, DTRAC, and GLOBAL, will be generated. Under the directory, two associated namelist files that include the controlling parameters (numerical or physical) are created. Besides, the needed source codes will be extracted from the template files and then stored in the directory src. It should be noted that the grid coupling case, i.e. NPBTAI, will have another two directories, i.e. NPB and TAI. In fact, the major preprocessing procedure is first carried out for each domain in the directories NPB and TAI, respectively. Subsequently, the script in the NPBTAI directory will form the combined namelist files required for the main code.

MAIN :

MAIN indicates the main code system. Following the same classification structure in PREP, the template and CASENAME directories exist here. The former covers all the template files and the later stands for a specific case. Note that four subdirectories appear in CASENAME. They are (i) src, (ii) INPUT_CASENAME, (iii) TEMP_CASENAME, and (iv) OUTPUT_CASENAME. Similarly, src includes the necessary source codes for the specified case. In INPUT_CASENAME, all the required initial input files generated by the preprocessor can be found. TEMP_CASENAME is used to save temporary files during model simulation. The location of OUTPUT_CASENAME is to store model results which are written in the NetCDF format. For the two-domain (or multi-)domain) case, the last three subdirectories are slightly different. They are divided according to their DOMAINNAMEs, e.g. INPUT_NPB and INPUT_TAI for the case NPBTAI.

POST :

POST means the postprocessor system. In TIMCOM, both MATLAB and NCL scripts are provided for each case to view model results. The subdirectories for different cases with two postprocessors can be found under the POST directory.

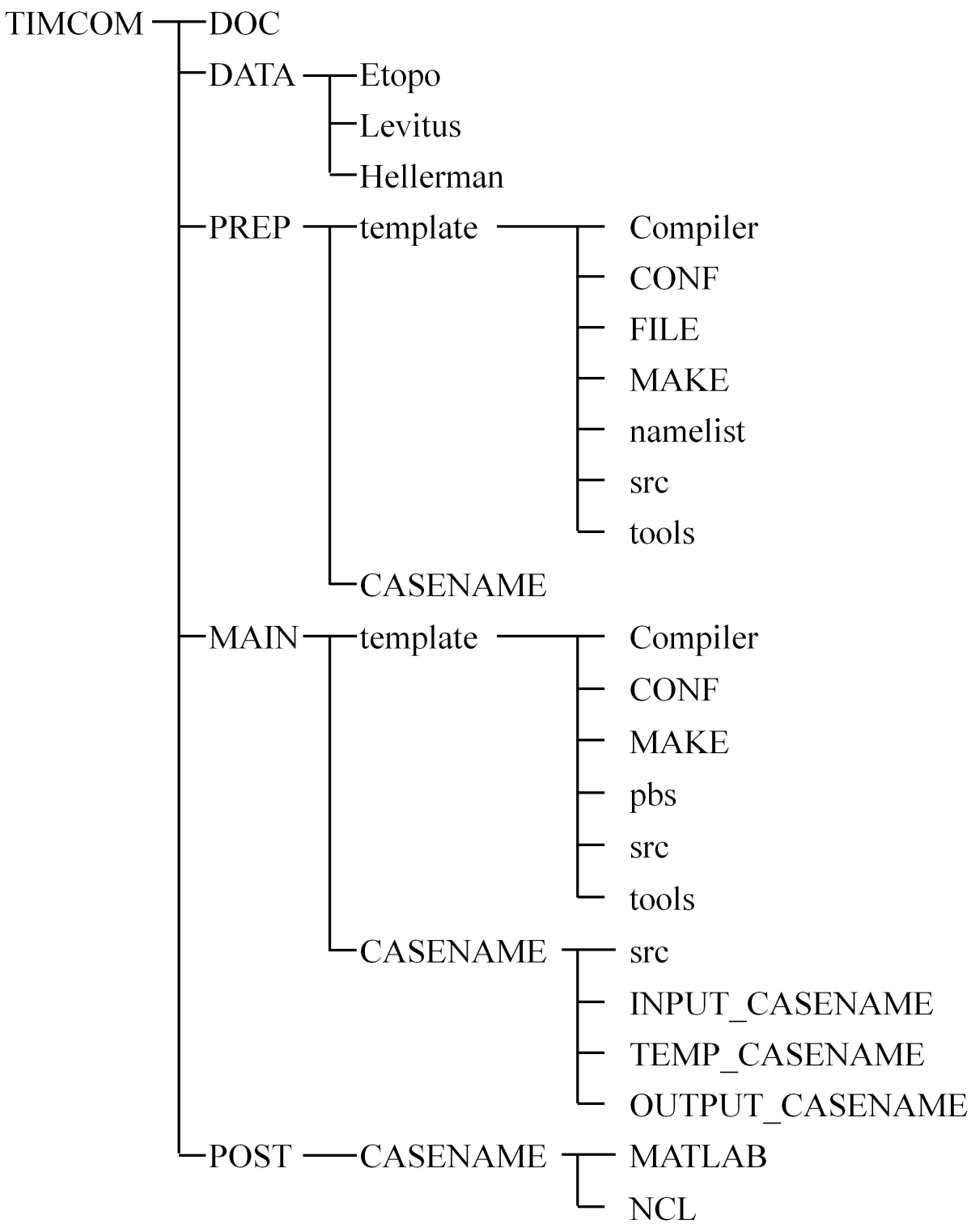

Figure 4.1: Program directories.

4.2 Source Codes

The descriptions for major source codes associated with (i) Variabl Modules, (ii) Preprocessor, (iii) Main Code, and (iv) Postprocessor are presented in this section.

4.2.1 Variable Modules

Based upon the purposes or functions, variables are declared and defined using different modules in several source files. In the following, significant source files are explained.

ocn_para.f90 :
The resolution parameters of TIMCOM, i.e. I_0, J_0 and K_0, are declared and defined in this file. Resolution constants $I_1 = I_0 - 1$, $J_1 = J_0 - 1$, and $K_1 = K_0 - 1$ are also defined.

grid_var_CASENAME.f90 :
Variables on the computation grid are declared and defined here, e.g. the dimensions of control volume, cell-averaged velocities, face-averaged velocities, pressure, temperature, salinity, density, and vertical viscosity/diffusivity. Mask arrays that identify land or sea cells are also declared.

control_var_CASENAME.f90 :
Flags that control features on/off (e.g. the effects of inflow at the lateral boundary) are declared and the associated physical/numerical parameters (e.g. inflow velocity) are also included.

4.2.2 Preprocessor

The purpose of the preprocessor system is to generate the initially required information such as grid coordinate, bathymetry map, climatology data, etc. These input files will be further described in chapter 5. Here, three major source codes and their functions are presented.

metgen_CASENAME.f90 :
Information of grid coordinate, e.g. spatial increments and locations of grid centers/lines, is generated through several subroutines in this file. Particularly notice that in the z direction, non-uniform grid spacing with exponential/simple stretching or user's definition can be adopted (by setting the option flag ZOP in the namelist file). Detail formulations for the non-uniform vertical layers can be found in Appendix A.

indata_CASENAME.f90 :
The source code file indata_CASENAME.f90 contains five major functions for general applications. They are briefly described as follows.

(1) Create bottom topography using interpolated Etopo5 bathymetry data.

(2) Calculate mask arrays, e.g. mask array KB indicates the location of bottom layer (equivalent to the number of active layers) in a given water column and another mask array IN identifies land/sea cells.

(3) Specify initial temperature and salinity fileds from Levitus climatology data.

(4) Impose surface wind forcing by Hellerman climatology data.

(5) Deal with lateral inflow/outflow boundaries

Note that control flags are used to switch to user-defined treatments (e.g. implement idealized bathymetry or temperature) or skip unnecessary processes (e.g. neglect wind effects) for simplified cases.

prep_CASENAME.f90 :
Subroutine INITFS in the source code file prep_CASENAME.f90 will (1) determine the initial vertical viscosity/diffusivity and (2) calculate the system matrix for the EVP elliptic pressure solver.

4.2.3 Main Code

The main code system solves hydrodynamics. Diagnostic function is employed to ensure no unexpectedly large velocity during the whole computation. Grid-coupling block will be adopted for the cases with two (or more) domains. Descriptions for major source codes are given in the following.

timcom_CASENAME.f90 :
The subroutine timcom_CASENAME controls the overall program flow, e.g. compute hydrodynamics, release tracer for particle tracking, perform diagnostics, output model results, etc.

initfs_CASENAME.f90 :
The subroutine initfs_CASENAME reads necessary input data to initialize the main code system for computation.

fs_CASENAME.f90 :
The computation of hydrodynamics is done in this file. Recall that TIMCOM solves the governing equations and boundary conditions using the second-order Leap-frog scheme and fourth-order approximation in time and space, respectively. The numerical algorithm includes three major steps: (i) calculate intermediate flow fields, (ii) determine pressure adjustment with EVP solver and update final flow field, and (iii) apply temporal filter. Detailed procedures to implement these three steps are illustrated in Figure 4.2.

wind_CASENAME.f90 :
The subroutines in wind_CASENAME.f90 deal with the surface wind forcings.

pmove_CASENAME.f90 :
The source code pmove_CASENAME.f90 is the implementation of particle tracking.

4.2.4 Postprocessor

TIMCOM provides two kinds of postprocessors, i.e. MATLAB and NCL scripts, to plot model results for better visualization. Comments are added in the scripts so that the users can easily understand and use them.

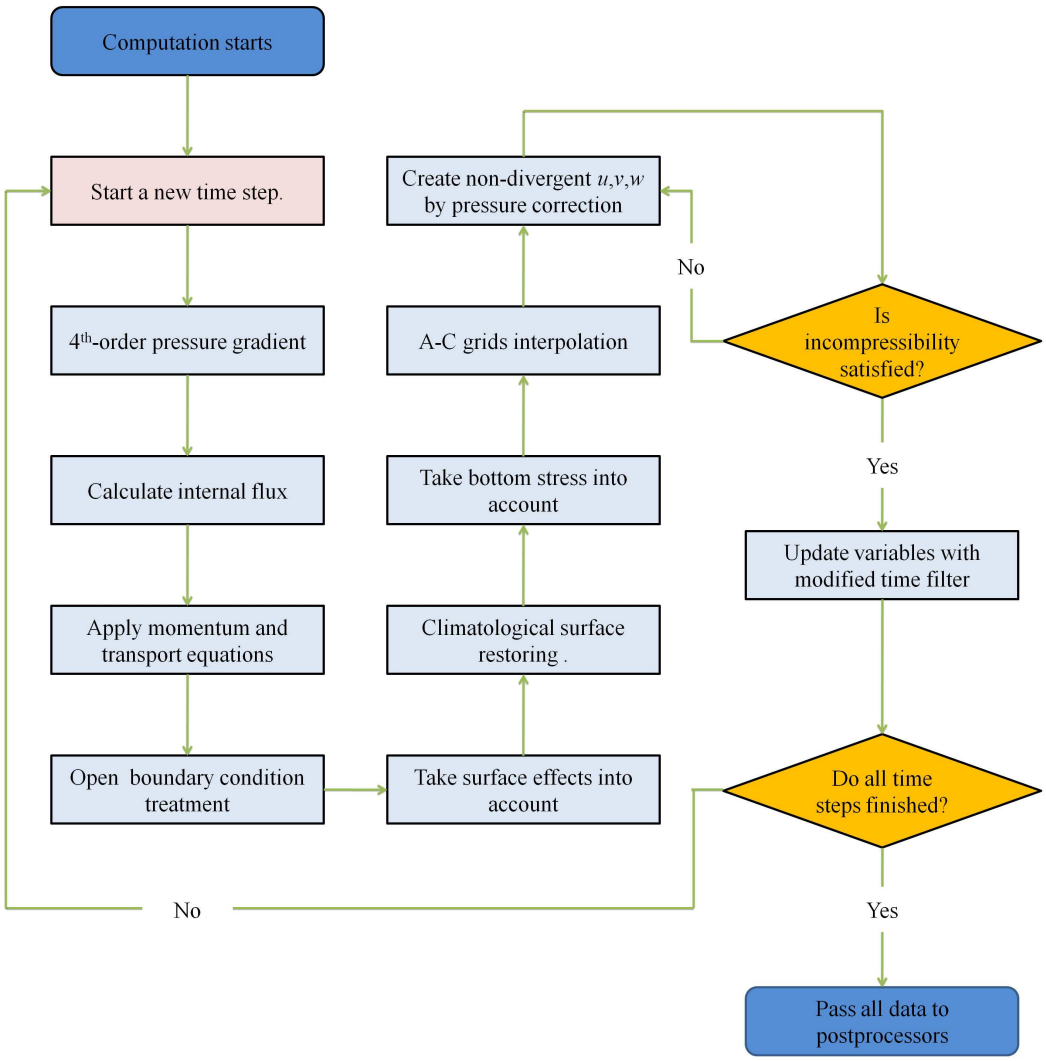

Figure 4.2: The computation procedures of TIMCOM.

Chapter 5

Model Applications and Tutorials

TIMCOM has been carefully examined and validated through several benchmark tests and real applications. We here illustrate four study cases including (i) 2D island wakes (denoted as ILE2D), (ii) 3D density current over a continental slope (denoted as DTRAC), (iii) eddy-permitting global ocean modelling (denoted as GLOBAL), and (iv) dual-grid north Pacific Ocean modelling (denoted as NPBTAI). Each case is indicated by different case number (see Table 4.1). In the following, the procedures to obtain the code, run the model, and plot the results are presented. Particularly, step-by-step tutorials for these cases are provided to help the users get familiar with the model.

Table 4.1 Each case and its corresponding case number in TIMCOM

Case Number	Case Name
1	ILE2D
2	DTRAC
3	GLOBAL
4	NPBTAI

5.1 Obtain the Code

Download TIMCOM (TIMCOM_v1.0.tgz) and the necessary data (TIMCOM_DATA.tgz) via our website "http://efdl.as.ntu.edu.tw/research/timcom/index.html". As a courtesy, we ask you to send us an email at yhtseng@as.ntu.edu.tw so that we can keep track of who's using the model in what application.

After downloading, put these two files under the user's directory "/home/username" and unzip them.

"tar -xzvf TIMCOM_v1.0.tgz" (Type "mv TIMCOM_v1.0 TIMCOM" for a shortened name.)

"tar -xzvf TIMCOM_DATA.tgz"

TIMCOM supports intel (default) and pgi (optional) fortran compiler in X86_64 Linux system. Notice that the NetCDF library (netcdf401.tgz) is required and its location can be specified using the command " export NETCDF = your NetCDF directory".

5.2 2D Island Wakes

5.2.1 Description

The oceanic flow around small islands has been extensively investigated (Dietrich *et al.*, 1996). A number of experimental and numerical works show that the flow behavior depends on Reynolds number. At very low Reynolds number, it is a creeping flow. At somewhat higher Reynolds numbers, two symmetrical standing vortices are formed but remain attached to the cylinder. At still higher Re, these vortices are stretched and wavy behavior of the tail can be observed. At even higher Re, alternating vortex shedding called the Karman vortex street is found. Besides, the results for a flat bottom confirm that the Coriolis terms strongly affect pressure but not the flow. TIMCOM is used to simulate this 2D Island Wake problem (ILE2D). In this case, the width and length of the domain are both 400 km. The water depth is 2 km. The inflow velocity, e.g. 40 cm/s, is specified at the south boundary. The grid resolution is 80x80 and the time step Δt is 1200 s. The model results well agree with the laboratory flows for both attached and shedding wake regimes.

5.2.2 Running the Model and Plotting the Results

Run "./timcom_setup 1" in the directory TIMCOM. Then, the required files for the case ILE2D will be automatically created according to the user's choice (simply press "y"). In the following, we will demonstrate how to carry out model simulations through running the preprocessor and main code systems as well as how to plot the results using the postprocessors. Note that the instructions below are mainly for the Linux users. At current stage, Windows users need to configure the Fortran source codes for each case in Linux and download them to the relevant directories.

Preprocessor

First, run the preprocessor to generate initial files for the main code system. The steps to run the preprocessor are:

1. Enter the PREP/ILE2D directory using "cd PREP/ILE2D".
2. Generate the Fortran 90 source codes for the preprocessor using "./configure".
3. Compile the preporcessor by the command "make", yielding the executable file "prep". If the user prefers to use pgi compiler, just type "make compiler=pgi" instead.
4. Modify the physical/numerical parameters in the file "namelist.tps_ILE2D" if needed.

   According to the case description above, we now have

   DIMEN = 2 (horizontal 2D case);

   TLX = TLY = 400.E5 (width and length of the domain are both 400 km);

   TLZ = 2.E5 (water depth is 2 km);

   LOPENS = LOPENN = 2 (open boundaries);

LOPENW = LOPENE = 0 (closed boundaries);

FL_INFL_S = 1 (inflow is specified at the south boundary);

FL_INFL_N = -1 (outflow);

VINFLOW = 40 (The inflow velocity is 40 cm/s);

I0_ILE2D = 82 (80 + 2 ghost cells);

J0_ILE2D = 82 (80 + 2 ghost cells);

K0_ILE2D = 3 (min. value for the horizontal 2D case);

DAODT = 72 (Δt = 86400/DAODT = 1200 s; Dt in the source code represents 2 Δt).

5. Execute the preprocessor using "./prep".

   The preprocessor creates the necessary input files for the main code system and saves them in the directory TIMCOM/MAIN/ILE2D/INPUT_ILE2D. These initial files are listed as follows.

   (a) ZKB
   This file involves the information of z levels, inverse layer thickness, mask arrays for land/water classification, and depth

   (b) boundaries
   This file contains the data of inflow velocity.

   (c) RUNDATA
   This file provides the information of the computation grid, e.g. grid spacing and its inverse. Vertical turbulent viscosity and diffusivity coefficients are also included.

   (d) EVP
   This file stores the coefficients for Poisson pressure operator.

   In addition, two files, i.e. namelist.input_ILE2D and namelist.run, will be generated under the directory TIMCOM/MAIN/ILE2D.

Main Code

Second, go to the directory TIMCOM/MAIN/ILE2D and carry out the simulations. The steps to run the main code system are:

1. Enter the directory TIMCOM/MAIN/ILE2D using "cd ../../MAIN/ILE2D".
2. Generate the Fortran 90 source codes using "./configure".
3. Compile the main code by the command "make", creating the executable file "timcom".
4. Modify the parameters in the file "namelist.run" if needed.

   Note that two parameters RUNS and ISAV can be modified to control the simulation time and the frequency of saving model results to the OUTPUT_ILE2D directory. RUNS represents the total time steps (e.g. 1440) or total simulation days (e.g. -20) while ISAV indicates the time steps between two outputs. If ISAV is a negative value, e.g. -1, main code will output the results once per simulation day.

5. Execute the main code using "./timcom" or "./timcom > run & " for background process. Upon normal termination, i.e. reach 1440 time steps or 20 simulation days, main code will show "job is successfully completed".

6. Restart the main code using "./timcom 1" if needed.

   If there is a need to carry out model simulations up to 120 days, modify the parameter RUNS = -120 and then type "./timcom 1". The main code can resume the last work from $t = 20$ days, rather than $t = 0$.

Postprocessor

Two different postprocessors, i.e. MATLAB and NCL scripts, are provided to plot the model results.

MATLAB: MATLAB(2009) script plot_vel_vor.m in POST/ILE2D/MATLAB is used to plot the velocity field and vorticity contour for the island wake case. In this m file, function readnc reads the NetCDF outputs from MAIN/ILE2D/OUTPUT_ILE2D and function cal_vor calculates vorticity. Two flags, i.e. fl_vel and fl_vor, determine whether velocity vectors and vorticity contours are shown or not (0: no; 1: yes). The users can specify any period for demonstration, e.g. n_day_st=1 and n_day_end=20. Type plot_vel_vor to run the m file, giving an animation of 20-day simulation. Finally, the last image will be saved as a TIF (or EPS) file, e.g. PLOT_VEL_VOR_00020.tif. The predicted velocity fields and vorticity contours at day 20 and 120 are illustrated in Fig. 5.1.

NCL: An open source graphic viewer NCL also can be used. Details about NCL can be found via http://www.ncl.ucar.edu/. Here, a NCL script vector_vort_ILE2D.ncl in the directory POST/ILE2D/NCL is prepared to show the predicted velocity and vorticity. For example, plot the results at $t = 20$ days (i.e. read the file "TYPE1_00020.nc") and save the graphic named as "vect" in the "pdf" format (i.e. vect.pdf) by typing

ncl vector_vort_ILE2D.ncl 'f_n="TYPE1_00020.nc"' 'output="pdf"' 'output_f="vect"',

where "pdf" can be replaced with "x11" for screen output or "png" for the portable network graphics.

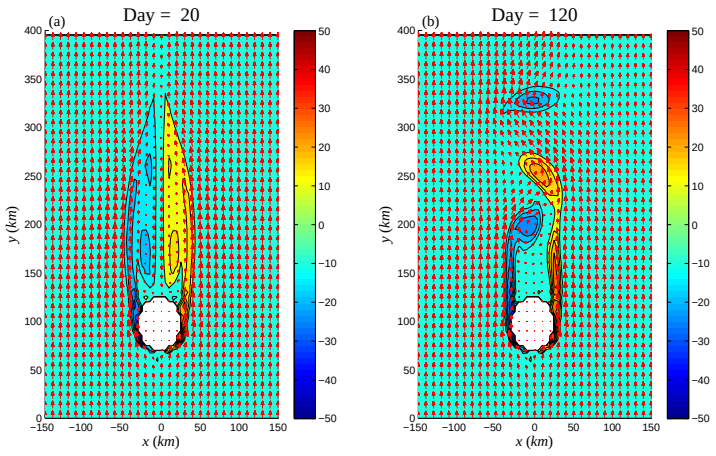

Figure 5.1: The velocity field and vorticity contour at day 20 and 120 at 80×80.

5.3 3D Density Current over a Continental Slope

5.3.1 Description

Modeling dense bottom density currents emanating from sill overflows is of major research interest. TIMCOM is applied to an idealized density current problem, i.e. the Dynamics of Overflow Mixing and Entrainment (DOME) experiment which is patterned after the Denmark Strait. Following the configuration in Ezer & Mellor (2004), the study region (see Fig. 5.2 (a)) is 1100 km (longitude) by 400 km (latitude). The minimum depth is 600 m at the northernmost latitude. A 1% shelf slope brings it to a maximum of 2160 m at 156 km offsore. Further south, the depth is uniformly 2160 m. In the model, a linear equation of state is used to specify the initial background temperature stratification (see Fig. 5.2 (b)). The bottom-concentrated and geostrophically balanced inflow with a total flow rate of 4.98 Sv (1 Sverdrup = $10^6 m^3/s$) is imposed at the northern boundary from x = -200 to -300 km. The bottom (top) layer inflow temperature linearly increases from 5.3 °C (13 °C) on its west side to 9.9 °C (15 °C) on its east side. The southern and eastern boundaries are closed. The western boundary is open, using an upwind-based, convective-type outflow condition to ensure zero net inflow. The computational domain is discretized by a set of uniform cells. The horizontal and vertical grid resolutions are 5 km and 60 m, respectively. The time increment Δt is 300 s for a total 30-day simulation time. Horizontal viscosity (v) and diffusivity (k) are assumed to be constant for simplicity (v = 50 and k = 10 m^2/s). Vertical viscosity and diffusivity are determined by a modified Richardson-number-based approach with background values of 0.1 cm^2/s (Pacanowski & Philander, 1981). These conditions give typical cell Reynolds number $O(100)$. For detailed model results and viscosity sensitivity, the users can refer to Tseng & Dietrich (2006).

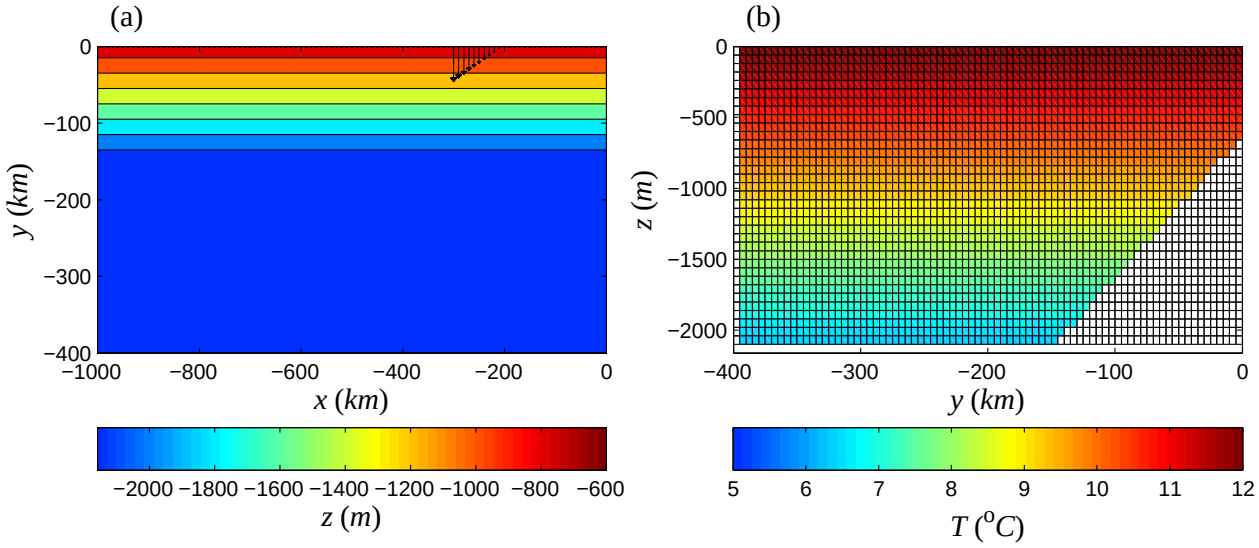

Figure 5.2: (a) The horizontal domain and northern inflow velocity vector at the bottom. (b) Illustration of latitudinal/vertical grid and initial temperature stratification.

5.3.2 Running the Model and Plotting the Results

The procedure to run the 3D density current case (denoted as DTRAC) is exactly the same except for the case number 2. In the root directory TIMCOM, type "./timcom_setup 2" to generate the necessary source files for the preprocessor and main code systems. We list all the steps again to help the users get familiar with TIMCOM.

Preprocessor

The steps to run the preprocessor are:

1. Enter the PREP/DTRAC directory using "cd PREP/DTRAC".
2. Generate the Fortran 90 source codes for the preprocessor using: "./configure".
3. Compile the preprocessor by the command "make", generating the executable file "prep".
4. Modify the physical/numerical parameters in the file "namelist.tps_DTRAC" if needed.
   According to the case description above, we now have
   DIMEN = 3 (3D case);
   TLX = 1100.E5 (length along longitudinal direction);
   TLY = 400.E5 (width along latitudinal direction);
   TLZ = 2160.E2 (maximum depth);
   LOPENW = LOPENN = 2 (open boundaries);
   LOPENE = LOPENS = 0 (closed boundaries);
   FL_INFL_N = 1 (inflow is specified at the north boundary);
   FL_INFL_W = -1 (outflow);
   FL_TS_ON = 1 (temperature is simulated);
   I0_DTRAC = 222 (220 + 2 ghost cells);
   J0_DTRAC = 82 (80 + 2 ghost cells);
   K0_DTRAC = 37 (36 layers bounded by 37 interfaces);
   ZOP = 3 (user-defined uniform vertical layers);
   DAODT = 288 (Δt = 86400/DAODT = 300 s).
5. Execute the preprocessor using "./prep".
   The initial files are created in the directory TIMCOM/DTRAC/INPUT_DTRAC.
   (a) ZKB - Z-level information, mask arrays, and depth
   (b) initial - Initial temperature
   (c) boundaries - Area of the west boundary cross-section and boundary velocities
   (d) RUNDATA - Information of the grid and vertical viscosity/diffusivity coefficients

(e) EVP - EVP pressure solver coefficients

Two namelist files, i.e. namelist.input_DTRAC and namelist.run, are generated within the directory TIMCOM/MAIN/DTRAC.

Main Code

Go to the directory TIMCOM/MAIN/DTRAC to run the code by the following steps.

1. Enter the directory TIMCOM/MAIN/DTRAC using "cd ../../MAIN/DTRAC" .
2. Generate the Fortran 90 source codes using "./configure".
3. Compile the main code by the command "make", yielding the executable file "timcom".
4. Modify the parameters in "namelist.run" or in "namelist.input_DTRAC" if needed.
5. Execute the main code using "./timcom".

Postprocessor

Similarly, the model results can be demonstrated using two postprocessors, i.e. MATLAB and NCL scripts.

MATLAB: There are four scripts written in roughly the same manner but for different purposes. The script plot_xy_bt_ta_vel.m shows temperature anomaly and velocity field along the bottom layer in the x-y plane. The script plot_xz_t.m gives temperature variation along the coast (i.e. x-z plane). Scripts plot_yz_t.m and plot_yz_ta_ave.m demonstrate the temperature and its average anomaly from the nearshor to the offshore (i.e. y-z plane), respectively. Here, we illustrate the prediction of fully developed plume distribution in Fig. 5.3

NCL: Currently, NCL script vector_conc_DTRAC is used to show temperature anomaly and velocity field. Other NCL scripts will be supported later.

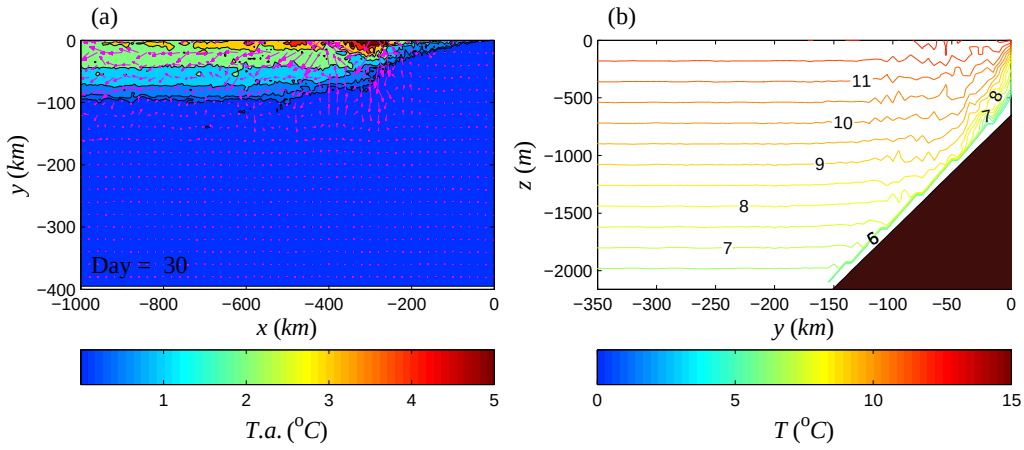

Figure 5.3: Prediction of fully developed plume distribution. (a) Bottom-layer temperature anomaly and velocity field and (b) temperature from the nearshor to the offshore.

5.4 Global Ocean Modelling

5.4.1 Description

Global ocean modelling is important to the study of ocean circulation and climate system. Particularly, accurate representation of multi-scale (either temporal or spatial) dynamics in the ocean is highly desirable and challenging. With successive progress in computer capacity and novel computational approaches, high-resolution modelling has become affordable.

Based upon the parallel version of TIMCOM (Tseng & Chien, 2011), the eddy-permitted global ocean simulation is carried out. The computation domain covers the entire globe from 72 °S to 72 °N with a closed northern boundary slowly nudging toward climatology in a sponge layer. The depth is derived from the unfiltered 5-min resolution bathymetry data (ETOPO5). To initialize the temperature and salinity fields, climatology of Levitus & Boyer (1994) is used. Density, as a function of temperature, salinity and pressure-depth, is determined by a nonlinear equation of state (Tseng *et al.*, 2005). The surface wind forcing is obtained from the interpolated monthly data of Hellerman & Rosenstein (1983). Besides, surface sources of heat and fresh water are specified by a non-damping approach of Dietrich *et al.* (2004a). Surface salinity is maintained by exchanges of salt with the second layer, ignoring minor effects of surface fresh water volume sources/sinks. In the model, a Mercator grid with 1/4 ° longitudinal resolution and 25 linear-exponentially stretched vertical layers are employed. The time increment Δt is 480 s. Vertical viscosity and diffusivity are determined by the approach of Pacanowski & Philander (1981).

In this document, we demonstrate the running procedure as well as 2 ° resolution results of TIMCOM. The efficient and accurate parallel version for high-resolution (i.e. 1/4 °) ocean modelling is available under license agreement. More results and detailed discussions can be found in the work of Tseng & Chien (2011).

5.4.2 Running the Model and Plotting the Results

Type "./timecom_setup 3" under root directory to create the preprocessor and main code systems for the GLOBAL case.

Preprocessor

The steps to run the preprocessor are:

1. Enter the PREP/GLOBAL directory using "cd PREP/GLOBAL".
2. Generate the Fortran 90 source codes for the preprocessor using "./configure".
3. Compile the preporcessor by the command "make", giving the executable file "prep".
4. Modify the physical/numerical parameters in the file "namelist.tps_GLOBAL" if needed.
   Based upon 2 ° resolution setup, we now have
   DIMEN = 3 (3D case);
   DXMNUT = 120 (120 minutes equivalent to 2 degree);

LOPENW = LOPENE = 1 (periodic boundaries);

LOPENS = 0 (closed boundary);

LOPENN = 2 (open boundary);

FL_RL_BATHY = 1 (real ocean bottom);

FL_ITPL = 1 (depth is interpolated from ETOPO5);

FL_LBS = 1 (lateral boundary shallowing treatment for the north boundary);

FL_TS_ON = 2 (both temperature and salinity fields are simulated);

FL_RL_TS = 1 (real temperature and salinity fields from Levitus climatology);

FL_WD_ON = 1 (surface wind forcing is imposed);

FL_RL_WD = 1 (real wind shear stress from Hellerman and Rosenstein);

I0_GLOBAL = 182 (180 uniform cells + 2 ghost cells);

J0_GLOBAL = 92 (90 non-uniform cells for maintaining grid aspect ratio + 2 ghost cells);

K0_GLOBAL = 31 (30 layers bounded by 37 interfaces);

ZOP = 1 (linear-exponentially stretched vertical layers);

ZTP = 1900.0 (top layer thickness is 19 m);

DAODT = 768 (Δt = 86400/DAODT = 112.5 s).

5. Execute the preprocessor using "./prep".

   The initial files are created in the directory TIMCOM/GLOBAL/INPUT_GLOBAL:

   (a) ZKB - Z-level information, mask arrays, and depth
   (b) initial - Initial temperature and salinity
   (c) boundaries - Surface conditions of temperature and salinity
   (d) winds - Wind shear stresses
   (e) TURBMIX - Background viscosity and diffusivity
   (f) RUNDATA - Information of the grid and vertical viscosity/diffusivity coefficients
   (g) EVP - EVP pressure solver coefficients

   Two namelist files (namelist.input_GLOBAL and namelist.run) are also generated under the directory TIMCOM/MAIN/GLOBAL.

Main Code

Go to the directory TIMCOM/MAIN/GLOBAL and run the model with the following steps.

1. Enter the directory TIMCOM/GLOBAL using "cd ../../MAIN/GLOBAL".
2. Generate the Fortran 90 source codes using "./configure".
3. Compile the main code using "make". The executable file "timcom" will be created.

4. Modify the parameters in the namelist files if needed.

5. Execute the main code

   We here demonstrate how to submit a job to a cluster. A shell file run.sh containing the scripts below is needed.

   #!/bin/bash -f

   #PBS -N GLOBAL

   #PBS -o GLOBAL.log

   #PBS -e GLOBAL.err

   #PBS -lnodes=1:ppn=1

   cd $PBS_O_WORKDIR

   ./timcom >output.txt

   The user can use the vi editor to create such a file (open the file "vi run.sh", enter edit mode "i", type the scripts above, and save the file ":wq"). Then change the mode of the shell file using "chmod u=rwx run.sh". This job now can be submitted by the command "qsub run.sh". To monitor the job status, simply type "qstat".

Postprocessor

MATLAB script plot_lonlat_ssh_t_vel.m or NCL script vector_contour_opt_GLOBAL.ncl can be used to show the yearly-averaged sea surface height and velocity field (see Fig. 5.4).

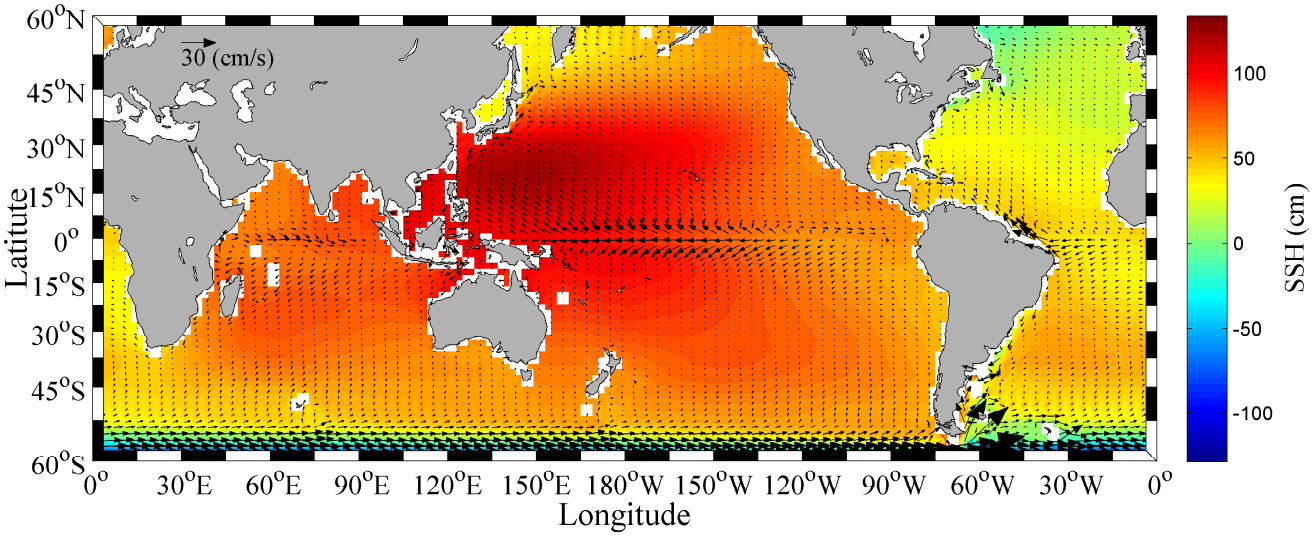

Figure 5.4: Predicted yearly-averaged sea surface height and velocity field.

5.5 Dual-domain North Pacific Ocean Modelling

5.5.1 Description

In the North Pacific, the major western boundary current Kuroshio has raised great research interests over the past several decades due to its significant influences on ship navigation, fisheries, marine resources, and global/regional ocean climate variations (Tseng *et al.*, 2011). TIMCOM based upon an efficient grid coupling framework (Dietrich *et al.*, 2004b) is capable of modelling the whole North Pacific Ocean (from 100 °E to 80 °W and from 30 °S to 60 °N, as shown in Fig. 3.1) with considerably reduced computational expense and providing accurate representations of the regional circulations in Asian Marginal Seas.

The dual-domain model for this study case (denoted as NPBTAI) consists of the North Pacific Basin domain (east of 150 °E in the right blue block with 1/4 ° resolution) and TAIwan area domain (west of 150 °E in left red block with 1/8 ° resolution). The depth is interpolated from the unfiltered 1-min resolution ETOPO-2 bathymetry supplemented by Taiwan's National Center for Ocean Research (Liu *et al.*, 1998). To initialize the temperature and salinity fields, seasonal climatology of Levitus & Boyer (1994) is used. The surface wind forcing is obtained from the interpolated monthly data of Hellerman & Rosenstein (1983). Besides, surface sources of heat and fresh water are determined by the non-damping approach of Dietrich *et al.* (2004a). The model domains are discretized based upon Mercator grid with different resolutions. Finer grid in the TAI domain adequately resolves ocean circulations in the marginal seas and coarser grid in the NPB domain maintains a minimum reasonable computation cost. In the vertical direction, both domains use 25 linear-exponentially stretched layers with a thickness of 6 m at the top layer. The lateral boundaries are closed except that the southern boundary of the NPB domain is slowly nudging toward climatology in a sponge layer. Note that the zonal boundary exchanges are specified by the energy conservative two-way coupling algorithm (see Chapter 3). The time steps are 200 s and 600 s for the TAI and NPB domains, respectively. In the model, the horizontal heat and momentum diffusivity coefficients are set to be 10^6 and 2×10^5 cm^2/s, respectively. Augmented molecular viscosity and diffusivity are determined by parameterized synoptic wind events and by breaking/non-breaking internal waves. The vertical mixing is parameterized by eddy diffusivity and viscosity using the formula of Pacanowski & Philander (1981). The users who are interested in the model results and discussions can refer to the work of Tseng *et al.* (2011).

5.5.2 Running the Model and Plotting the Results

Type "./timcom_setup 4" under root directory to create the preprocessor and main code systems for the NPBTAI case.

Preprocessor

Note that preprocesses for NPB and TAI domains need to be done separately, unlike those introduced in other single domain cases. Besides, the complete namelist files containing the information of both domains need to be further generated. The steps are listed as follows.

1. Enter the PREP/NPB directory using "cd PREP/NPB".

2. Generate the Fortran 90 source codes for the preprocessor using "./configure".
3. Compile the preporcessor ("make") to generate the executable file "prep".
4. Modify the physical/numerical parameters in the file "namelist.tps_NPB" if needed.
   According to the case description above, we now have
   DIMEN = 3 (3D case);
   DXMNUT = 15 (15 minutes equivalent to 1/4 degree);
   LOPENE = LOPENS = LOPENN = 0 (closed boundaries);
   LOPENW = 2 (open boundary for transferring values to TAI);
   FL_RL_BATHY = 1 (real ocean bottom);
   FL_LBS = 1 (lateral boundary shallowing treatment for the north boundary);
   FL_TS_ON = 2 (both temperature and salinity fields are simulated);
   FL_RL_TS = 1 (real temperature and salinity fields from Levitus climatology);
   FL_WD_ON = 1 (surface wind forcing is imposed);
   FL_RL_WD = 1 (real wind shear stress from Hellerman and Rosenstein);
   I0_NPB = 521 (519 uniform cells + 2 ghost cells);
   J0_NPB = 428 (426 non-uniform cells for maintaining grid aspect ratio + 2 ghost cells);
   K0_NPB = 26 (25 layers bounded by 37 interfaces);
   ZOP = 1 (linear-exponentially stretched vertical layers);
   ZTP = 600.0 (top layer thickness is 6 m);
   DAODT = 144 (Δt = 86400/DAODT = 600 s).
5. Execute the preprocessor for the NPB domain using "./prep".
6. Enter the PREP/TAI directory using "cd ../TAI".
7. Generate the Fortran 90 source codes for the preprocessor using "./configure".
8. Compile the preporcessor ("make") to generate the executable file "prep".
9. Modify the physical/numerical parameters in the file "namelist.tps_TAI" if needed.
   According to the case description above, we now have
   DIMEN = 3 (3D case);
   DXMNUT = 7.5 (7.5 minutes equivalent to 1/8 degree);
   LOPENW = LOPENS = LOPENN = 0 (closed boundaries);
   LOPENE = 2 (open boundary for transferring values to NPB);
   FL_RL_BATHY = 1 (real ocean bottom);
   FL_TS_ON = 2 (both temperature and salinity fields are simulated);

FL_RL_TS = 1 (real temperature and salinity fields from Levitus climatology);

FL_WD_ON = 1 (surface wind forcing is imposed);

FL_RL_WD = 1 (real wind shear stress from Hellerman and Rosenstein);

I0_TAI = 406 (404 uniform cells + 2 ghost cells);

J0_TAI = 488 (486 non-uniform cells for maintaining grid aspect ratio + 2 ghost cells);

K0_TAI = 26 (25 layers bounded by 37 interfaces);

ZOP = 1 (linear-exponentially stretched vertical layers);

ZTP = 600.0 (top layer thickness is 6 m);

DAODT = 342 (Δt = 86400/DAODT = 200 s).

10. Execute the preprocessor for the TAI domain using "./prep".
11. Enter the PREP/NPBTAI directory using "cd ../NPBTAI".
12. Generate the complete namelist file using "./configure".

The "./prep" steps generate the initial files for NPB and TAI. The last step yields the complete namelist files for NPBTAI.

Main Code

Run the main code with similar steps in the directory TIMCOM/MAIN/NPBTAI.

1. Enter the directory TIMCOM/MAIN/NPBTAI using "cd ../../MAIN/NPBTAI".
2. Generate the Fortran 90 source codes using "./configure".
3. Compile the main code by the command "make", generating the executable file "timcom".
4. Modify the parameters in the namelist files if needed.
5. Submit the job to execute TIMOCM.

Postprocessor

Similarly, with a given domain, MATLAB script plot_lonlat_ssh_t_vel.m as well as NCL script vector_contour_opt_GLOBAL.ncl can be used to demonstrate model results (see Fig. 5.5 and Fig. 5.6).

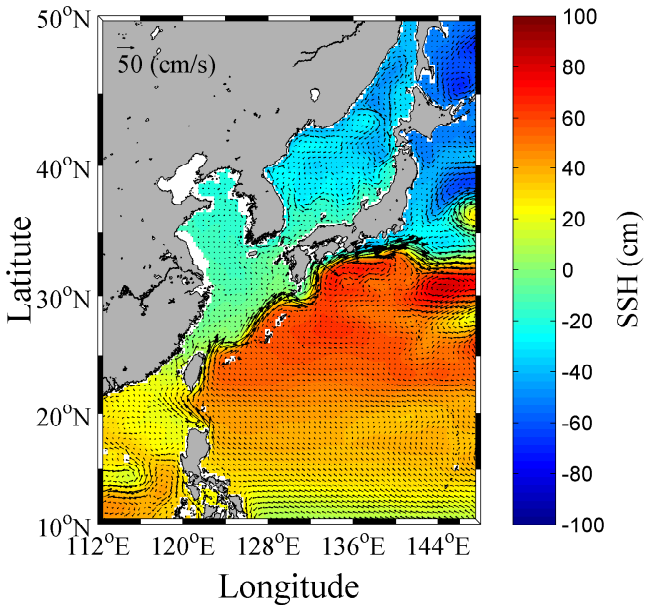

Figure 5.5: Predicted yearly-averaged sea surface height and velocity field of TAI domain.

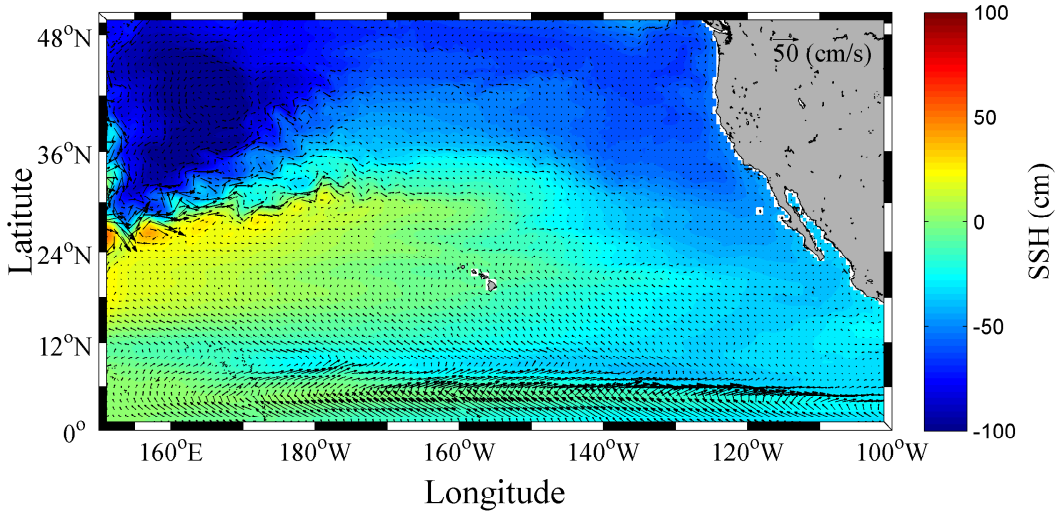

Figure 5.6: The same as Fig. 5.5 except for NPB domain.

Appendix A

Nonuniform Vertical Grid Spacing

For ocean circulation modelling, a finer grid resolution is usually needed at upper layers to accurately resolve the complicated vertical structure of flow field or near the bottom to well represent the topography. The uniform vertical grid spacing (see Fig. A.1 (a)) can be easily used to discretize the computational domain. However, the computation can be relatively expensive due to the large number of grids in the whole water column while a sufficiently high resolution is employed to capture detailed flow features. In contrast, nonuniform vertical grid spacing is an effective alternative to achieve the required accuracy with a reasonable computational expense. TIMCOM provides three different types of nonuniform grids, generating (i) linear-exponential stretched, (ii) power stretched, and (iii) user-defined vertical layers. Their corresponding values of the option flag **ZOP** are 1, 2, and 3, respectively. In the following, we briefly describe the vertical grid generation.

a) Vertical Grid Generation

In the model, K_1 (or K_0-1) vertical grids are generated by specifying the information of z levels, rather than the thickness of each layer. The z levels of the cell center and its bounded interfaces at layer k are denoted as z_{2k}, z_{2k-1}, and z_{2k+1}, respectively. Once the z levels are created, the model further calculates the distances between grid lines/centers and saves them in an inverse form for later computation.

$$1/\Delta z_k = 1/(z_{2k+1} - z_{2k-1}) \quad , k = 1, 2, \ldots K1$$
$$1/\Delta zc_k = 1/(z_{2k} - z_{2k-2}) \quad , k = 2, 3, \ldots K1$$

b) Linear-Exponential Stretched Vertical Layers (ZOP = 1)

Generally, an exponential stretching is employed in the z direction, concentrating vertical cells near surface (see Fig. A.1 (b)). In the model, a combination of linear and exponential functions is used to determine the z levels including the interfaces (odd indexes) and cell centers (even ones), i.e.

$$z_n = a + b \cdot exp\left[-c(n-1)\left(\frac{-h}{2K_1}\right)\right] + d(n-1)\left(\frac{-h}{2K_1}\right) \quad , n = 1, 2, \ldots, 2K_1, 2K_1 + 1$$

or

$$z_n = a + b \cdot exp\left[-c \cdot zu_n\right] + d \cdot zu_n \quad , n = 1, 2, \ldots, 2K_1, 2K_1 + 1$$

where n is the index of z-level array with $z_1 = 0$ (or $z_{2K_1+1} = -h$) at the surface (or bottom); $a = -(1-d)h/[1 - exp(ch)]$; $b = -a$; c and d are the user-specified parameters; h is the maximum water depth; and zu_n represents the uniform distributed z levels.

c) Power Stretched Vertical Layers (ZOP = 2)

Vertical stretching along the z direction can be determined by the power law as well. The users need to specify the z level of the top-layer cell $z_2 = -z_{TOP}$, giving the half thickness of the top layer $\Delta z_{half} = z_1 - z_2 = z_{TOP}$. As the depth gets deeper, the lower grids are stretched (or contracted) by a grid stretching ratio r, which will be determined in the program. The power stretched vertical layers (see Fig. A.1(c)) can be expressed as

$$z_n = z_{n-1} - \Delta z_{half} \cdot r^{(n-2)}, \qquad n = 3, 4, \ldots, 2K_1, 2K_1 + 1$$

d) User-Defined Vertical Layers (ZOP = 3)

The users can also define any arbitrary vertical layers (as shown in Figure A.1(d)) through the z-level data in the file **Z_CUSTOM_CASENAME.TXT**. We should note that the data array with a length of $2K_1 + 1$ would be required for K_1 layers and odd/even indexes represent the interfaces/cell centers.

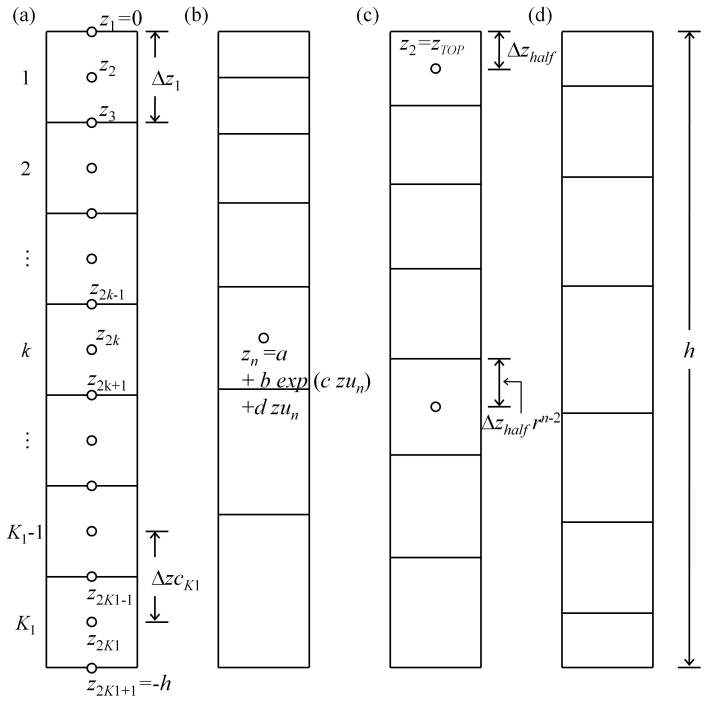

Figure A.1: (a) uniform, (b) linear-exponential stretched, (c) power stretched, and (d) user-defined vertical layers.

Appendix B

Vertical Mixing Scheme: Pacanowski & Philander (1981)

Vertical viscosity and diffusivity A_v and K_v are given by the formulae:

$$A_v = 0.02 + min(10, 100R^2) + R|W|dz/Re \tag{B.1}$$

$$K_v = 0.02 + Rmin(10, 100R^2) + R|W|dz/ReR \tag{B.2}$$

where R=1/(1+5Ri); Ri is the gradient Richardson number; $|W|$ is the magnitude of vertical velocity ; dz is the vertical grid interval; and Re is the specified vertical cell Reynolds number which is simply a non-dimensional ratio of advection to turbulent diffusion for variations having the scale of order one model grid interval.

The first term in Eqns (B.1) and (B.2) represents laminar diffusivity. The second one is a standard Pacanowski and Philander mixing parameterization. The third is a generally small mixed physical/numerical term, assigned a value of 10, which avoids cell Reynolds number overshoots by vertical advection, such as produced by internal waves. It allows extremely small vertical mixing necessary to maintain the cold intermediate layer throughout the model year. The physical processes represented by terms two and three are the vertical Reynolds stress terms representing the nonlinear interaction of subgrid-scale flow components with resolved scales. These diffusion terms are conventional ad hoc representations of the subgrid-scale effects which have limited basis outside of boundary layers.

Appendix C

Fourth-Order Accurate Interpolation from Nearby Control Volume Averages to a Cell Face

The fourth-order interpolation scheme is derived (see Figure C.1 for the schematic of the cells with no loss of generality). Only x-direction is demonstrated here for the purpose of simplicity and similar analysis can be applied to the other directions. At a given (y, z), let any variable q (e.g. temperature) be a third-degree polynomial:

$$q(x, y, z) = C_0(y, z) + C_1(y, z)x + C_2(y, z)x^2 + C_3(y, z)x^3 \tag{C.1}$$

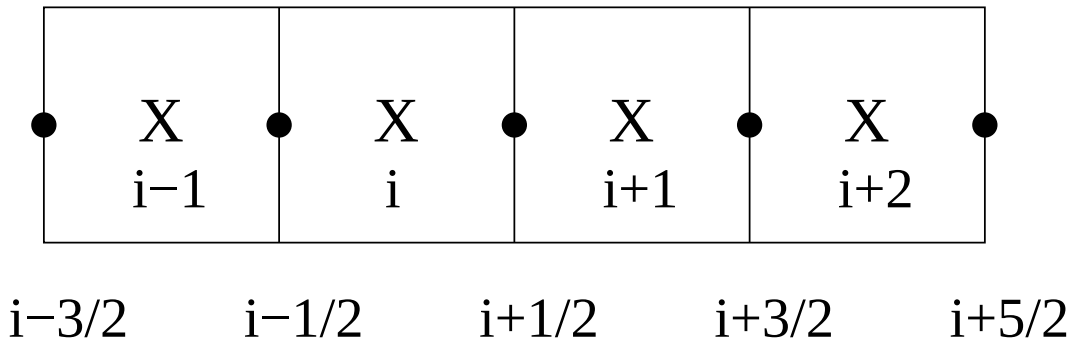

Figure C.1: Schematic of the cells and the index for the fourth-order interpolation.

Conveniently, let $x_{i+1/2} = 0$, so $q(0, y, z) = q(x_{i+1/2}, y, z) = C_0(y, z)$. We can define the quantities q at a time level.

$$\begin{aligned} q_{i+2}(y,z) &= \int_{x_{i+\frac{3}{2}}}^{x_{i+\frac{5}{2}}} q(x,y,z)dx \\ &= C_0(y,z)x|_{x_{i+\frac{3}{2}}}^{x_{i+\frac{5}{2}}} + \frac{1}{2}C_1(y,z)x^2|_{x_{i+\frac{3}{2}}}^{x_{i+\frac{5}{2}}} + \frac{1}{3}C_2(y,z)x^3|_{x_{i+\frac{3}{2}}}^{x_{i+\frac{5}{2}}} + \frac{1}{4}C_3(y,z)x^4|_{x_{i+\frac{3}{2}}}^{x_{i+\frac{5}{2}}} \end{aligned} \tag{C.2}$$

$$\begin{aligned} q_{i+1}(y,z) &= \int_{x_{i+\frac{1}{2}}}^{x_{i+\frac{3}{2}}} q(x,y,z)dx \\ &= C_0(y,z)x|_{x_{i+\frac{1}{2}}}^{x_{i+\frac{3}{2}}} + \frac{1}{2}C_1(y,z)x^2|_{x_{i+\frac{1}{2}}}^{x_{i+\frac{3}{2}}} + \frac{1}{3}C_2(y,z)x^3|_{x_{i+\frac{1}{2}}}^{x_{i+\frac{3}{2}}} + \frac{1}{4}C_3(y,z)x^4|_{x_{i+\frac{1}{2}}}^{x_{i+\frac{3}{2}}} \end{aligned} \tag{C.3}$$

$$\begin{aligned} q_{i}(y,z) &= \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} q(x,y,z)dx \\ &= C_0(y,z)x|_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} + \frac{1}{2}C_1(y,z)x^2|_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} + \frac{1}{3}C_2(y,z)x^3|_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} + \frac{1}{4}C_3(y,z)x^4|_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \end{aligned} \tag{C.4}$$

$$\begin{aligned} q_{i-1}(y,z) &= \int_{x_{i-\frac{3}{2}}}^{x_{i-\frac{1}{2}}} q(x,y,z)dx \\ &= C_0(y,z)x|_{x_{i-\frac{3}{2}}}^{x_{i-\frac{1}{2}}} + \frac{1}{2}C_1(y,z)x^2|_{x_{i-\frac{3}{2}}}^{x_{i-\frac{1}{2}}} + \frac{1}{3}C_2(y,z)x^3|_{x_{i-\frac{3}{2}}}^{x_{i-\frac{1}{2}}} + \frac{1}{4}C_3(y,z)x^4|_{x_{i-\frac{3}{2}}}^{x_{i-\frac{1}{2}}} \end{aligned} \tag{C.5}$$

Solving the above equations for the interfacial value $C_0(y, z)$, we get

$$C_0(y,z) = q_{i+1/2}(y,z) = \frac{7}{12}(q_i(y,z) + q_{i+1}(y,z)) - \frac{1}{12}(q_{i-1}(y,z) + q_{i+2}(y,z)) \tag{C.6}$$

The coefficients can also be obtained by Taylor series expansion of variable $q(x, y, z)$ along $x-$axis and integrating the cell analytically to get the discretized cell averaged

$$q_i(y,z) = \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} q(x,y,z)dx \tag{C.7}$$

which yields

$$\begin{aligned} q_{i+1/2}(y,z) &= q_i(y,z) + q_{i+1}(y,z) + \frac{1}{4}q''(8)(\Delta x)^3 \\ &= q_{i-1}(y,z) + q_{i+2}(y,z) + \frac{1}{4}q''(56)(\Delta x)^3 \end{aligned} \tag{C.8}$$

Then use Richardson extrapolation to obtain the above equation (C.6). Finally, integrate over the control volume face $\sigma(y, z)$ to obtain cell face average,

$$\begin{aligned} \iint_\sigma q_{i+1/2}(y,z)dydz = \frac{7}{12}\iint_\sigma (q_i(y,z) + q_{i+1}(y,z))dydz \\ -\frac{1}{12}\iint_\sigma (q_{i-1}(y,z) + q_{i+2}(y,z))dydz \end{aligned} \tag{C.9}$$

or in the notation form:

$$\overline{q}_{i+1/2,j,k} = \frac{7}{12}(Q_{i,j,k} + Q_{i+1,j,k}) - \frac{1}{12}(Q_{i-1,j,k} + Q_{i+2,j,k}) \tag{C.10}$$

Sanderson & Brassington (1998) obtained the above face-averaged quantity $\overline{q}_{i+1/2,j,k}$ using a cumulative sum of the cell-averaged quantity. The current derivation is complete and can be extended to non-uniform grids when the expansion ratio is known. Figure C.2 shows the truncation error of various differencing operators through the example that a sinusoidal signal is numerically differentiated. The error is plotted as a function of the ratio of grid spacing Δ to wavelength λ. The power-law scaling of error with the ratio λ/Δ is closely followed for wavelengths greater than 4Δ with C-grid differencing and for wavelengths greater than 6Δ with A-grid. Further, the scaling applies with reasonable approximation to the smallest wavelengths resolvable on the grid. The error analysis and numerical examples in Sanderson & Brassington (1998) demonstrate that the fourth-order interpolation used in TIMCOM is fourth-order accurate on both A-and C-grids.

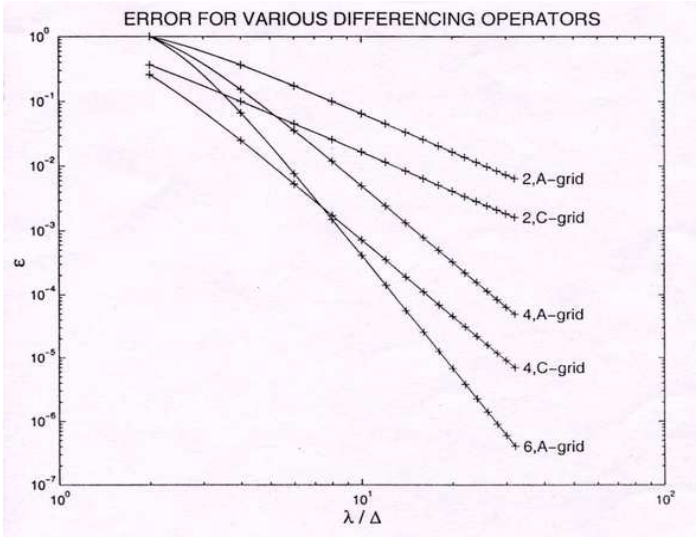

Figure C.2: Error ε for differentiating s sinusoidal function using the A-grid (collocated) and C-grid (staggered) differencing operators. The error is plotted as a function of the ratio of grid spacing Δ to wavelength λ. The label 2,A-grid refers to the second-order operator on a collocated grid. The label 2,C-grid refers to the second-order operator on a staggered grid. Similarly, label 4,A-grid refers to the fourth-order operator on a collocated grid and label 4,C-grid refers to the fourth-order operator on a staggered grid. The sixth-order differencing operator on the A grid is labeled 6,A-grid. [Figure 1 of Sanderson & Brassington (1998)].

Appendix D

Error Vector Propagation Method

Generally, one can obtain a Poisson equation for the pressure $p_{i,j}$ in ocean general circulation models, i.e.

$$\frac{p_{i+1,j} - 2p_{i,j} + p_{i-1,j}}{\Delta x_i^2} + \frac{p_{i,j+1} - 2p_{i,j} + p_{i,j-1}}{\Delta y_j^2} = r_{i,j}, \tag{D.1}$$

where Δx and Δy are the horizontal grid spacings of each cell. Error Vector Propagation (EVP) method can be used to solve the equation efficiently. The basic idea is to obtain the error $e_{i,j}$ for correcting the intermediate pressure field $p'_{i,j}$. Note that pressures at the boundaries are known and their associated errors are zero (i.e. $e_{i,1} = e_{1,j} = e_{I_0,j} = 0$) except for the north boundary error. The algorithm of EVP method contains three major steps and will be briefly described in the following.

First, guess the pressure at one grid inside the south boundary, e.g. $p'_{i,2} = p_{i,1}$, and the unknown error vector is $E_s = e_{i,2} = p_{i,2} - p'_{i,2}$. Rearranging Eq. (D.1) and substituting the initial guess gives the expression for the remaining pressures and the error propagation equation, i.e.

$$p'_{i,j+1} = \Delta y^2 r_{i,j} - \frac{\Delta y^2}{\Delta x^2} p'_{i-1,j} + 2(1 + \frac{\Delta y^2}{\Delta x^2}) p'_{i,j} - \frac{\Delta y^2}{\Delta x^2} p'_{i+1,j} - p'_{i,j-1}. \tag{D.2}$$

$$e_{i,j+1} = -\frac{\Delta y^2}{\Delta x^2} e_{i-1,j} + 2(1 + \frac{\Delta y^2}{\Delta x^2}) e_{i,j} - \frac{\Delta y^2}{\Delta x^2} e_{i+1,j} - e_{i,j-1}. \tag{D.3}$$

The intermediate pressures can be directly calculated from $j = 2$ up to $j = J_1$ (see Fig. D.1), thus propagating the unknown errors forward and resulting in the errors at the north boundary $E_n = e_{i,J_0} = p_{i,J_0} - p'_{i,J_0}$.

Second, solve the unknown south boundary error vector E_s, i.e.

$$[C]_{I_2 \times I2} \cdot [E_s]_{I_2 \times 1} = [E_n]_{I_2 \times 1}\,, \tag{D.4}$$

where $[C]$ is the influence coefficient matrix. Each column of the matrix $[C]$ indicates the response to a unit error impose at the south boundary and can be established using the propagation equation (D.3). In the solution procedure, matrix $[C]$ is the same while time is marching. Thus, at each time step, E_s (or $e_{i,2}$) can be efficiently solved by the LU decomposition method.

Finally, apply Eq. (D.3) again to obtain the interior errors and the exact solution can be fully updated. The EVP solution is second-order accurate with respect to the grid size since central difference is employed.

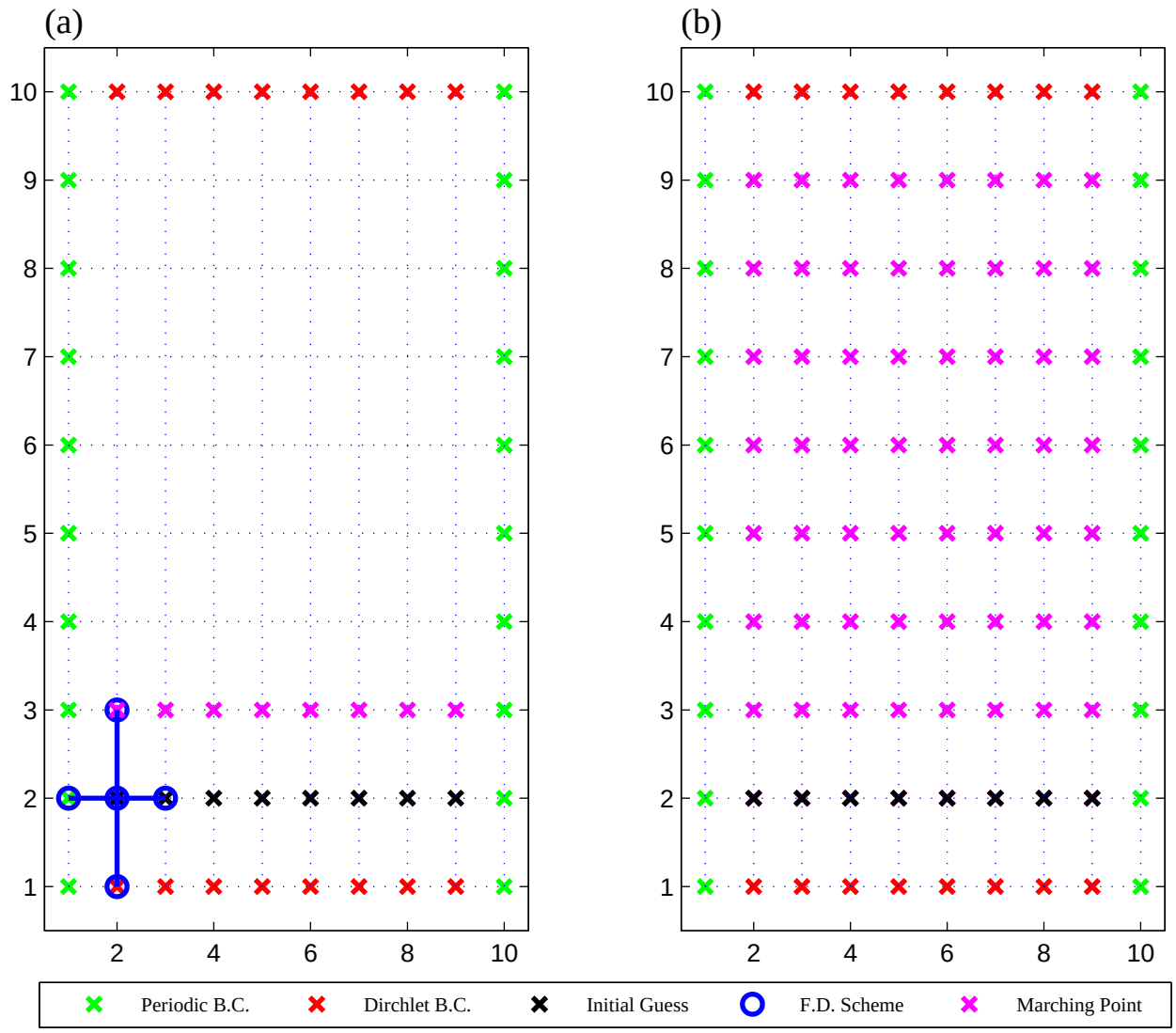

Figure D.1: (a) the calculation of intermediate pressures using Eq. (D.2) and (b) the estimated pressures after one sweep (i.e. $j = 2$ to $j = J_1$).

Appendix E

Listing of Variables

This list includes only variables of significance. Local variables are used mainly for scratch or are secondary in nature. The ones with asterisks preceded are user-specified quantities, while others are derived by the preprocessor or the TIMCOM model. Many variables are used only for diagnostics.

A: Total area of wet points in each layer.

AB: Left(west) coefficient in Poisson pressure operator.

AC: Central coefficient in Poisson pressure operator.

AB: Bottom(south) coefficient in Poisson pressure operator.

AR: Right(east) coefficient in Poisson pressure operator.

AT: Top(north) coefficient in Poisson pressure operator.

AROUT: Area of boundary region to be adjusted to get zero net inflow.

AV: Current number of time levels in long term average.

AVGXYT: Horizontal mean temperature and salinity data.

*B: Coefficient of thermal expansion for water in equation of state.

*CS,CSV: Spherical coordinate latitude cosine factors.

*DAODT: Number of time steps per model day.

*DAYS: Present time in days.

*DE0: Background horizontal diffusivity.

DHX,DHY: X, Y directed turbulent diffusivity.

*DM0: Background horizontal viscosity.

DMX,DMY: X, Y directed turbulent viscosity.

DMZ0: Background vertical viscosity.

*DRAG: Bottom drag coefficient.

DT: Time step size.

DUM0,DUM1,DUM2: Scratch arrays.

DX,DY,DXV,DYV: Lateral grid increments across cell-centers and grid lines.

EV: Vertical turbulent viscosity array.

EVAP: Integrated E-P volume flux.

F: Coriolis parameter.

*FLTW: Coefficient in fltw time marching method of Asselin filter.

FNEW,FOLD: Time interpolation factors between months NEW,NLD.

*G: Gravity.

GB: Gravity times coefficient of thermal expansion.

H: Scratch array used by surface pressure Poisson solver.

HV: Vertical turbulent diffusivity array.

*IE: Array of evp block boundaries used in Poisson pressure solver.

IN: Mask array for scalar quantities.

INFX,INFY: Boundary mask arrays for biharmonic filter.

*IPLOT: Time step between main graphics data outputs.

IT0: First time step of present run, equals to zero or number of restart time.

ITF: Time step number.

ITFDAY: Number of time steps per day.

IU0: 2-D mask array for staggered X component of advection velocity, applies to top layer only.

IU: 3-D mask array for staggered X component of advection velocity.

IV0: 2-D mask array for staggered Y component of advection velocity, applies to top layer only.

IV: 3-D mask array for staggered Y component advection velocity.

IW: 3-D mask array for staggered Z component advection velocity.

KB: 2-D array, number of levels containing water in logical depth.

*KTRM: Thermocline level, used for animation graphics only.

*LHEAT: Flag for surface heating of air-sea exchange.

*LMOVI: Flag to save data for animation.

*LOPEN: Flag for open boundary conditions.

*LRSTRT: Flag to initialize from restart file.

*LSAVE: Flag to save restart file.

*LSOL: Flag for radiative heating of vertical profile.

*LTURB: Flag for call turbulence submodel subroutine.

*LWIND: Flag to include wind forcing.

*M1,M2,M3,M4,M5,M6,M7,M8,M5M: Miscellaneous output frequency controls.

*MVI: Number of time steps between movie frame storage.

*MXIT: Last time step of present run. Time step number is saved and not reset to zero when restart occurs.

*MXPLOT: ¡aximum number of time steps between main graphics data outputs.

N360: Present time in days after start of present model year.

NEW,NLD: Present, previous month in annual cycle.

NSOMBO: Number of ensemble-averaging months to present.

OCS,OCSV: Spherical coordinate latitude inverse cosine factors.

ODX,ODY,ODV,ODXV,ODT: Inversed horizontal grid increments and time step.

ODZ: Inversed layer thickness array, the distance between W-levels.

ODZW: Inversed distance between layer pressure levels.

OFLTW: Unity minus FLTW.

OLDE, OLDQ: Previous months E-P and surface Q.

ORZMX: Inversed of specified vertical cell Reynolds number.

P0: Pressure against rigid lid or, hydrostatically surface height.

P: Pressure anamoly, the deviation from base hydrostatic.

PBAR: Time mean surface pressure.

PRN: Prandtl number , $DM0/DE0$.

PVAR: RMS deviation of surface pressure from its time mean.

PX,PY: Scratch arrays for horizontal pressure gradient.

QAVG: Multi-year ensemble monthly time mean surface heating.

QPROF: Vertical radiative heating profile array.

QSUM: Mean surface layer heating, $watts/m - m$.

RHO: Potential density deviation from minimum potential density.

RINV1: Influence coefficient array for elliptic pressure solver.

RINV: Inversed array for elliptic pressure solver.

RMSV: Mean squared velocity anomaly at depth of $700m$.

RNN: Array used to blend 2nd and 4th order p.g. near boundaries.

RZMX: Specified vertical cell Reynolds number.

S1,S2,SLF: Old, filtered (central), and leapfrog salinity fields.

S: Source term for rigid lid pressure Poisson equation.

SBAR: Time mean salinity.

SCLI: Annual mean climatological salinity.

SCR: 3-D multi-purpose scratch array.

SSURF,TSURF: Monthly surface climatology salinity and temperature.

SSURFM, TSURFM: Horizontal mean monthly surface climatology.

SUMIN: Number of 'wet points' in each model layer.

SZ: Z component of salinity flux, for advection only.

TANPHI: Tangent of latitude.

T1,T2,TLF: Old, filtered (central), and leapfrog T fields.

TAUDT: Surface layer restoring (nudging) factor.

*TAU: Surface restoring time in days.

*TAUN: Lateral boundary restoring time in days.

TAVG: Horizontal mean temperature.

TBAR: Time mean temperature.

TCLI: Annual mean climatological temperature.

TLZ: Maximum depth of modeled domain. Deeper water depth is set to TLZ.

TNUDGE: Summed surface layer temperature nudging during present month.

TSP: Longitudinally averaged monthly temperature climatology.

TW: Running time averages used in vertical heat flux diagnostics.

TX: X component of total (advection + diffusion) heat flux.

TY: Y component of total (advection + diffusion) heat flux.

TZ: Z component of heat flux, for advection only.

U1,U2,ULF: Old, filtered (central), and leapfrog X velocity fields.

U: Staggered 'C' grid X component of non-divergent advection velocity.

UCLI: Time mean longitudinal velocity at depth of $700m$.

UX: X component of total (advection + diffusion) U momentum flux.

UY: Y component of total (advection + diffusion) U momentum flux.

UZ: Z component of U momentum flux, for advection only.

V1,V2,VLF: Old, filtered (central), and leapfrog Y velocity fields.

V: Staggered 'C' grid Y component non-divergent advection velocity.

VCLI: Time mean latitudinal velocity at depth of $700m$.

VX: X component of total (advection + diffusion) V momentum flux.

VY: Y component of total (advection + diffusion) V momentum flux.

VZ: Z component of V momentum flux, for advection only.

W: Vertical velocity

WATTS: Conversion factor from (°C per time step) to W/w^2.

WAVG: Multi-year ensemble monthly time mean precipitation-evaporation.

XBAR: Time mean streamfunction for vertically averaged flow.

Y: Latitude cell centers.

YV: Latitude grid lines.

Z: Stretched vertical grid array includes both cell center and face levels.

Bibliography

Dietrich, D. E. 1997. Application of a modified 'a' grid ocean model having reduced numerical dispersion to the gulf of Mexico circulation. *Dyn. Atmos. Oceans*, **27**, 201–217.

Dietrich, D. E., & Ko, D. S. 1994. A semi-collocated ocean model-based on the SOMS approach. *Int. J. Numer. Methods Fluids*, **19**, 1103–1113.

Dietrich, D. E., & Lin, C. A. 2002. Effects of hydrostatic approximation and resolution on the simulation of convective adjustment. *Tellus*, **54(A)**, 34–43.

Dietrich, D. E., Marietta, M. G., & Roache, P. J. 1987. An ocean modeling system with turbulent boundary layers and topography: Part 1. Numerical studies of small island wakes in the ocean. *Int. J. Numer. Methods Fluids*, **7**, 833–855.

Dietrich, D. E., Bowman, M. J., Lin, C. A., & Mestas-Nunez, A. 1996. Numerical studies of small island wakes in the ocean. *Geophys. Astrophys. Fluid Dyn.*, **83**, 195–231.

Dietrich, D. E., Haney, R. L., Fernandez, V., Josey, S., & Tintore, J. 2004a. Air-sea fluxes based on observed annual cycle surfce climatology and ocean model internal dynamics: A precise, non-damping zero-phase-lag approach applied to the Mediterranean Sea. *J. Mar. Systems*, **52**, 145–165.

Dietrich, D. E., Mehra, A., Haney, R. L., Bowman, M. J., & Tseng, Y. H. 2004b. Dissipation effects in modeling the North Atlantic ocean. *Geophys. Res. Lett.*, **31**, L05302.

Ezer, T., & Mellor, G.L. 2004. A generalized coordinate ocean model and a comparison of the bottom boundary layer dynamics in terrain-following and in z-level grid. *Ocean Modelling*, **6**, 379–403.

Hellerman, S., & Rosenstein, M. 1983. Normal monthly wind stress over the world ocean with error estimates. *J. Phys. Oceanogr.*, **13**, 1093–1104.

Levitus, S., & Boyer, T. P. 1994. *World Ocean Atlas, Vol. 4: Temperature. NOAA Atlas NESDIS 4.* 117 pp.

Liu, C. S., Liu, S. Y., Lallemand, S. E., Lumdberg, N., & Reed, D. L. 1998. Digital elevation model offshore Taiwan and its tectonic implications. *Terr. Atmos. Ocean Sci.*, **9**, 705 – 738.

Pacanowski, R. C., & Philander, S. G. H. 1981. Parameterization of vertical mixing in numerical models of tropical ocean. *J. Phys. Oceanogr.*, **11**, 1443–1451.

Sanderson, B. G., & Brassington, G. 1998. Accuracy in the context of a control-volume model. *Atmos.-Ocean*, **36**, 355–384.

Tseng, Y. H., & Chien, M. H. 2011. Parallel Domain-decomposed Taiwan Multi-scale Community Ocean Model (PD-TIMCOM). *Computers & Fluids*, **45**, 77–83.

Tseng, Y. H., & Dietrich, D. E. 2006. Entrainment and transport in idealized three-dimensional gravity current simulation. *J. Atmos. Oceanic Technol.*, **23**, 1249–1269.

Tseng, Y. H., Dietrich, D. E., & Ferziger, J. H. 2005. Regional circulation of the Monterey Bay region- Hydrostatic versus Non-hydrostatic modeling. *J. Geophys. Res.*, **110**, doi:10.1029/2003JC002153.

Tseng, Y. H., Jan, S., & Shen, M. L. 2011. Validation of the Kuroshio Current system in the dual-domain Pacific ocean model framework. *Progress in Oceanography*, **accepted**.

Williams, P. D. 2009. A proposed modificaton to the Robert-Asselin time filter. *Monthly Weather Review*, **137**, 2538–2546.